

XIII Congreso del Máster en Investigación Matemática y Doctorado en Matemáticas



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



VNIVERSITAT
DE VALÈNCIA

**Facultad de Matemáticas
Universitat de València
12, 13 y 14 de Enero de 2026**

Editado por Juan R. Torregrosa y Francisco I. Chicharro

XIII Congreso del Máster Universitario en Investigación Matemática y Doctorado en Matemáticas

Burjassot, 12 a 14 de enero de 2026



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



VNIVERSITAT
DE VALÈNCIA

Este libro incluye los resúmenes extendidos de los trabajos presentados en el XIII Congreso del Máster Universitario en Investigación Matemática y Doctorado en Matemáticas, coorganizado por la Universitat Politècnica de València y la Universitat de València.

ISBN: 978-84-09-89822-0

Editado por Juan R. Torregrosa Sánchez y Francisco I. Chicharro López

I.U.I. Matemática Multidisciplinar

Universitat Politècnica de València



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



VNIVERSITAT
DE VALÈNCIA

Este libro incluye los resúmenes extendidos de los trabajos presentados en el XIII Congreso del Máster Universitario en Investigación Matemática y Doctorado en Matemáticas, coorganizado por la Universitat Politècnica de València y la Universitat de València.

Entre las actividades que lleva a cabo el Máster Universitario en Investigación Matemática, se encuentra el congreso anual del Máster Universitario en Investigación Matemática y Doctorado en Matemáticas, que tiene lugar en el Salón de Grados de la Facultad de Matemáticas de la Universitat de València.

Este congreso ofrece la oportunidad de que estudiantes de máster y doctorado puedan presentar sus trabajos de investigación, intercambiando ideas con expertos en las diferentes áreas de investigación y mejorando sus habilidades a la hora de presentar y exponer trabajos en público.

El XIII Congreso del Máster Universitario en Investigación Matemática y Doctorado en Matemáticas ha sido coorganizado por la Universitat Politècnica de València y la Universitat de València.

Comité Organizador

- Enric Cosme Llop  , Universitat de Val  ncia
- Isabel Cordero Carri  n, Universitat de Val  ncia
- Juan Ram  n Torregrosa S  nchez, Universitat de Val  ncia

Índice general

A Physiological Model of Exercise-Induced Glucose Regulation in Women with Type 1 Diabetes <i>M. T. Mendoza, B. Ricarte, J. L. Díez, and J. Bondia</i>	1
Family of optimal derivative-free iterative schemes <i>Julissa H. Jerezano, Francisco I. Chicharro, and Neus Garrido</i>	8
Framework for computing the Moore-Penrose tensor inverse <i>J. Pérez-Alemaný, J. C. Azcona-Leiva, and A. Cerdán Martínez</i>	15
Frontal surfaces and their singularities <i>C. Muñoz-Cabello</i>	28
Iterative methods for approximating the Moore-Penrose inverse of complex matrices <i>Juan Carlos Azcona Leiva, Aarón Cerdán Martínez, and Joan Pérez Alemany</i> .	35
Iterative Methods for the Drazin Inverse of a Matrix with Real and Complex Spectrum <i>Aarón Cerdán Martínez, Juan Carlos Azcona Leiva, and Joan Pérez Alemany</i> .	52
Morphisms of rewriting systems <i>Juan Climent Vidal, Enric Cosme Llópez, and Raúl Ruiz Mora</i>	66
PINNs and PINOs for solving PDEs, review and implementation <i>Adrián Pereira-García, Guillem Rodilla-Alamà, and Wenceslao Tafalla-Domínguez</i>	84
Qualitative study of a tumor growth model <i>Cristina Vilar-Medina, M. Carmen Martí, and Antonio Baeza</i>	104
The Hidden Frequencies of Functions: The Mathematics of Harmonic Analysis <i>Irene Buffi and Gaia Mereu</i>	113
Three unified Farey-sequence applications for rational direction discretization in digital image analysis <i>Harold Díaz-Iza, Bushra Rani, and Ana Sáez-Morillo</i>	126

A Physiological Model of Exercise-Induced Glucose Regulation in Women with Type 1 Diabetes

M. T. Mendoza^{*1}, B. Ricarte¹, J.L.Díez², and J. Bondia²

¹*Department of Applied Mathematics, Universitat Politècnica de València, Camí de Vera s/n, 46022-València, Spain*

²*Department of Systems and Automatic Engineering, Universitat Politècnica de València, Camí de Vera s/n, 46022-València, Spain*

Abstract

Physical exercise plays a central role in metabolic control in individuals with Type 1 Diabetes Mellitus (T1D), although it also represents a major risk factor for hypoglycemia. In this work, a dynamic physiological model, termed HGM-Ex-T1D, is proposed to describe exercise-induced glucose regulation in women with T1D by integrating exercise intensity, muscle metabolism, and exogenous insulin administration.

The model is inspired by the structure of the Hovorka model but is explicitly adapted to exercise conditions through the inclusion of lactate as a metabolic mediator and by considering insulin as an external control input. The final objective is to provide an interpretable mathematical framework that allows simulation, analysis, and validation of real exercise scenarios in women with T1D using experimental data.

Keywords: physical exercise, type 1 diabetes, physiological modeling, glucose regulation, insulin.

Introduction

Glucose regulation during physical exercise in individuals with Type 1 Diabetes Mellitus is a complex physiological process driven by the interaction between exercise-induced muscle glucose uptake, hepatic glucose production, and exogenous insulin administration [3, 4]. In women, this regulation may be further modulated by hormonal factors and increased metabolic variability during exercise [5, 6].

Classical mathematical models, such as the Hovorka model, have been widely used

^{*}mamenbal@mat.upv.es

for glucose control applications in Type 1 Diabetes [1, 2], but present limitations when applied to real exercise conditions, particularly due to the implicit treatment of physical activity and the absence of muscle metabolic mediators. In this context, the present work proposes a physiologically grounded extension of this approach, specifically adapted to exercise and validated using experimental data in women.

Materials

The HGM-Ex-T1D model is conceived as a low-dimensional dynamic system designed to capture the dominant mechanisms governing exercise-induced glucose regulation in women with T1D. The model integrates three coupled subsystems: muscle glycolytic metabolism, hepatic glucose production, and the action of exogenous insulin.

Model variables

The dynamic state variables of the model are:

- $G(t)$: blood glucose concentration.
- $L(t)$: blood lactate concentration.
- $I(t)$: effective plasma insulin concentration.

The model inputs are defined as:

- $u_E(t)$: exercise intensity.
- $u_I(t)$: administered insulin.

Exercise intensity definition

Exercise intensity is defined using a normalized heart rate formulation:

$$u_E(t) = \frac{FC(t) - FC_{\text{rest}}}{FC_{\text{max}} - FC_{\text{rest}}}$$

where $FC(t)$ denotes the measured heart rate at time t , FC_{rest} the resting heart rate, and FC_{max} the maximal heart rate estimated or experimentally observed.

This normalization provides a dimensionless input bounded between 0 and 1, facilitating its use as a physiological proxy for metabolic workload.

Initial conditions

Initial conditions are defined as follows:

- $G(0)$ corresponds to basal pre-exercise blood glucose.

- $L(0) = 1$ mmol/L represents a basal metabolic state.
- $I(0)$ is estimated from the daily basal insulin dose:

$$I(0) = \frac{\text{daily basal insulin dose (IU)}}{24}$$

This assumes a quasi-steady-state insulin condition prior to exercise onset.

Mathematical model

The dynamics of effective plasma insulin are described by:

$$\frac{dI(t)}{dt} = -k_I I(t) + u_I(t)$$

Lactate dynamics are modeled as:

$$\frac{dL(t)}{dt} = k_L^{prod} \max(0, u_E(t) - u_{LT}) - k_L^{clear} L(t)$$

Hepatic glucose production is defined as:

$$EGP(t) = EGP_{basal} e^{-\alpha_I I(t)} + k_{EGP} u_E(t)$$

Muscle glucose uptake is given by:

$$U_m(t) = k_{U0} + k_{UI} I(t) + k_{UE} u_E(t) + k_{UL} L(t)$$

Finally, blood glucose dynamics are described by:

$$\frac{dG(t)}{dt} = EGP(t) - U_m(t)$$

Simulation experiments

Model validation is conducted through the following simulation experiments:

- Reproduction of real exercise sessions using experimental inputs.
- Sensitivity analysis to basal insulin dose variations.
- Identification of critical exercise intensity thresholds.
- Comparison with a reduced model excluding lactate-dependent uptake.

Numerical implementation

The HGM-Ex-T1D model was implemented in Python and numerically integrated using an explicit Euler scheme with fixed time step. A simplified implementation of the model dynamics is shown below.

```
import numpy as np

def HGM_Ex_T1D(state, uE, uI, params, dt):
    G, L, I = state

    # Unpack parameters
    kI = params['kI']
    kLprod = params['kLprod']
    kLclear = params['kLclear']
    uLT = params['uLT']
    EGPb = params['EGP_basal']
    alphaI = params['alphaI']
    kEGP = params['kEGP']
    kU0 = params['kU0']
    kUI = params['kUI']
    kUE = params['kUE']
    kUL = params['kUL']

    # Insulin dynamics
    dI = -kI * I + uI

    # Lactate dynamics
    dL = kLprod * max(0, uE - uLT) - kLclear * L

    # Hepatic glucose production
    EGP = EGPb * np.exp(-alphaI * I) + kEGP * uE

    # Muscle glucose uptake
    Um = kU0 + kUI * I + kUE * uE + kUL * L

    # Glucose dynamics
    dG = EGP - Um

    # Euler integration
    G += dG * dt
    L += dL * dt
    I += dI * dt

    return np.array([G, L, I])
```

Listing 1: Python implementation of the HGM-Ex-T1D model

Results

Table 1 summarizes the main simulation results obtained with the proposed HGM-Ex-T1D model for a representative subset of women with Type 1 Diabetes. The table reports the root mean square error (RMSE) between simulated and experimental glucose profiles, as well as the minimum glucose values observed during exercise sessions.

Table 1: Main simulation results obtained with the HGM-Ex-T1D model.

Subject	Session	RMSE (mg/dL)	$G_{\min}$ Exp. (mg/dL)	$G_{\min}$ Sim. (mg/dL)
W1	Session 1	12.4	78	82
W1	Session 2	14.1	74	76
W2	Session 1	10.9	85	88
W2	Session 2	13.3	81	83

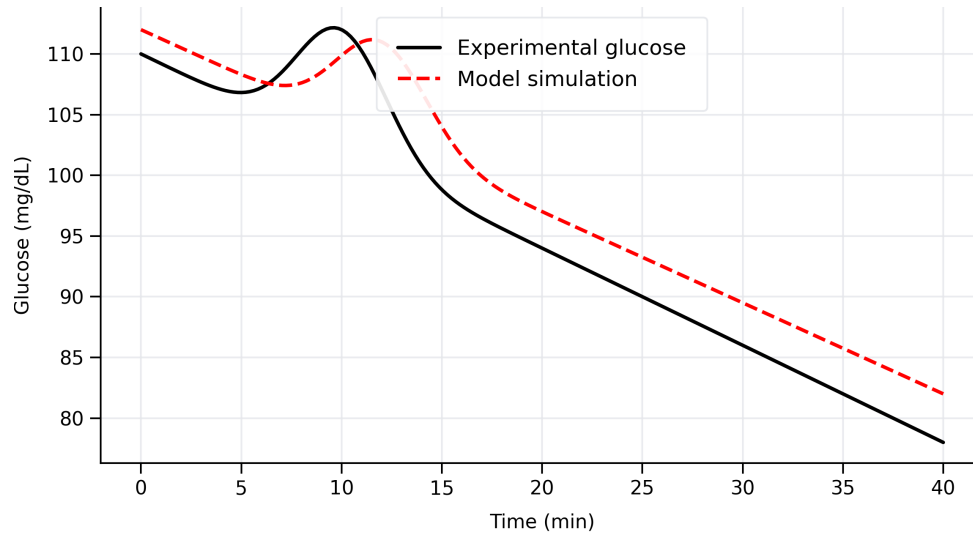
The results indicate that the model is able to reproduce the overall glucose dynamics observed experimentally, including exercise-induced glucose declines and minimum glucose values during high-intensity phases.

Figure 1 illustrates the institutional framework of the study and the academic context in which the model was developed.

Figure 1: Institutional framework of the study.



Figure 2: Experimental and simulated glucose profiles during an exercise session. The HGM-Ex-T1D model reproduces the overall glucose dynamics and exercise-induced glucose decline observed experimentally.



As shown in Figure 2, the proposed model captures the main trends of glucose dynamics during exercise, including the decrease in glucose levels associated with higher exercise intensity. In addition to session reproduction, sensitivity analyses showed that variations in basal insulin input produced physiologically consistent changes in simulated glucose profiles, with increased insulin levels leading to larger glucose declines during exercise. The inclusion of lactate-dependent glucose uptake significantly improved model performance in high-intensity scenarios.

Conclusions

In this work, a physiologically grounded dynamic model for exercise-induced glucose regulation in women with Type 1 Diabetes has been proposed and implemented. The HGM-Ex-T1D model integrates exercise intensity, muscle metabolism, hepatic glucose production and exogenous insulin administration within an interpretable mathematical framework.

Simulation results demonstrate that the model is capable of reproducing experimental glucose trends and capturing hypoglycemia-prone exercise conditions. The explicit inclusion of lactate as a metabolic mediator represents a key contribution of the model, allowing improved representation of high-intensity exercise responses.

The proposed framework constitutes a solid basis for future model refinement, validation with larger datasets and potential applications in personalized exercise planning for women with Type 1 Diabetes.

Acknowledgements

This study was supported within the academic framework of the Universitat Politècnica de València and developed in collaboration with institutional research initiatives in the field of mathematical modeling and applied physiology. This work was also supported by the research project PID2022-1377230B-C21, under the framework project “*Glucose control in women versus men towards sex-specific insulin therapies in patients with Type 1 Diabetes Mellitus*”.

References

- [1] R. Hovorka, V. Canonico, L. J. Chassin et al., *Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes*, *Physiological Measurement*, 25(4), 905–920, 2004.
- [2] R. Hovorka et al., *A model of glucose regulation in type 1 diabetes mellitus*, *Biomedical Engineering, IEEE Transactions on*, 51(5), 199–208, 2004.
- [3] D. H. Wasserman, H. B. Davis, *Exercise, hepatic metabolism, and insulin action*, *American Journal of Physiology-Endocrinology and Metabolism*, 296(5), E1049–E1056, 2009.
- [4] B. B. Rasmussen, C. A. Volpi, *Muscle protein metabolism in exercise and insulin resistance*, *Advances in Nutrition*, 2(1), 28–38, 2011.
- [5] A. L. Hicks, *Gender differences in glucose metabolism and exercise*, *Sports Medicine*, 40(4), 279–292, 2010.
- [6] M. J. Joyner, E. F. Barnes, *Sex differences in exercise physiology*, *Physiology*, 30(6), 413–423, 2015.

Family of optimal derivative-free iterative schemes

Julissa H. Jerezano^{*1,2}, Francisco I. Chicharro³, and Neus Garrido³

¹PhD program in Mathematics, Universitat Politècnica de València, Camí de Vera s/n, 46022-València, Spain.

²Departamento de Matemática, Universidad Nacional Autónoma de Honduras, 21102-San Pedro Sula, Honduras.

³I.U. Matemática Multidisciplinar, Universitat Politècnica de València, Camí de Vera s/n, 46022-València, Spain.

Abstract

It is very common in mathematics and engineering to solve nonlinear equations, which are often difficult or impossible to solve analytically. Therefore, it becomes necessary to use iterative methods that, under certain conditions, allow us to obtain approximate solutions. In recent years, numerous strategies have been developed for this purpose. General purpose iterative techniques for solving nonlinear equations provide approximations for problems without closed form solutions. However, these methods lose some of their properties when the problems involve multiple roots or nondifferentiable functions, in which case specific methods are required. In this work, a new family of optimal derivative free iterative methods for finding multiple roots is proposed. Specifically, a two step iterative scheme with a weight function is designed, where the first step is based on a modified Traub-Steffensen method. The resulting methods achieve fourth order convergence without using derivatives. Numerical results confirm that the proposed derivative-free methods are competitive with other robust methods available in the literature, including both derivative based and derivative-free methods.

Introduction

The problem of finding the roots of a nonlinear equation is one of the oldest problems in numerical approximation [1]. In the present work, iterative methods are considered for finding a multiple root α of multiplicity $m \geq 1$ of a nonlinear equation $g(x) = 0$, that is,

$$g(\alpha) = g'(\alpha) = g''(\alpha) = \cdots = g^{(m-1)}(\alpha) = 0, \quad g^{(m)}(\alpha) \neq 0.$$

One of the most widely used methods for finding a multiple root of $g(x)$ is the

^{*}jhjerull@doctor.upv.es

modified Newton's method, which has quadratic convergence [2]:

$$x_{k+1} = x_k - m \frac{g(x_k)}{g'(x_k)}, k = 0, 1, 2, \dots \quad (1)$$

In addition to method (1), there exist in the literature many methods with different orders of convergence for finding multiple roots of $g(x) = 0$. Most of these methods require the evaluation of first-order derivatives or both first- and second-order derivatives. In this sense, one of the objectives of this work is to design derivative-free methods. Another objective is for these methods to be optimal in the sense of the Kung-Traub conjecture [3], which states that the order of convergence p of a memory-less method performing d distinct functional evaluations per iteration satisfies

$$p \leq 2^{d-1},$$

and a method is called optimal when equality holds.

Derivative-free methods are useful when the derivative of the function g difficult to compute, expensive to evaluate or it does not exist. One of the derivative-free techniques is the Traub-Steffensen-type scheme [4], which replaces g' in Newton's iterative method by the approximation

$$g'(x_k) \simeq \frac{g(x_k + \beta g(x_k)) - g(x_k)}{\beta g(x_k)} = g[w_k, x_k],$$

where

$$g[w_k, x_k] = \frac{g(w_k) - g(x_k)}{w_k - x_k}$$

is the first-order divided difference with

$$w_k = x_k + \beta g(x_k), \quad \beta \in \mathbb{R} \setminus \{0\}.$$

Thus, the modified Newton's method (1) is transformed into the modified Traub-Steffensen's method:

$$x_{k+1} = x_k - m \frac{g(x_k)}{g[w_k, x_k]}, k = 0, 1, 2, \dots \quad (2)$$

The modified Traub-Steffensen's method (2) has an advantage over the modified Newton's method (1), since it preserves quadratic convergence without requiring derivatives.

In this work, a family of iterative schemes for solving nonlinear equations with multiple roots is presented. The design of this family is based on the use of weight functions. Section 2 is devoted to describing the construction procedure of the proposed family. Section 3 presents the numerical experiments with three nonlinear functions. Finally, Section 4 summarizes the main conclusions of this work.

Methods

Recently, Kumar et al. [5] proposed the following optimal second-order scheme:

$$x_{k+1} = x_k - G(\theta), \quad k = 0, 1, 2, \dots \quad (3)$$

where $G(\theta) : D \subset \mathbb{C} \rightarrow \mathbb{C}$ is a weight function and

$$\theta = \frac{g(x_k)}{g[w_k, x_k]}.$$

Starting from the iterative scheme with a weight function (3) and choosing

$$w_k = x_k + \beta[g(x_k)]^2, \quad \beta \in \mathbb{C} \setminus \{0\},$$

we propose a multipoint iterative scheme composed of two steps of the form:

$$\begin{aligned} y_k &= x_k - m \frac{g(x_k)}{g[w_k, x_k]}, \\ x_{k+1} &= x_k - m H(t_k) \frac{g(x_k)}{g[w_k, x_k]}, \quad k = 0, 1, 2, \dots \end{aligned} \quad (4)$$

where $m \geq 1$ denotes the multiplicity of the required root, and $H(t)$ is a single-variable weight function. Moreover,

$$g[w, x] = \frac{g(w) - g(x)}{w - x}.$$

In addition,

$$t = \left(\frac{g(y_k)}{g(x_k)} \right)^{1/m}$$

is a multivalued function. We consider its principal branch defined as

$$t = \exp \left\{ \frac{1}{m} \log \left(\frac{g(y_k)}{g(x_k)} \right) \right\},$$

where

$$\log \left(\frac{g(y_k)}{g(x_k)} \right) = \log \left| \frac{g(y_k)}{g(x_k)} \right| + i \arg \left(\frac{g(y_k)}{g(x_k)} \right), \quad -\pi < \arg(z) \leq \pi.$$

Equivalently,

$$t = \left| \frac{g(y_k)}{g(x_k)} \right|^{1/m} \exp \left\{ \frac{i}{m} \arg \left(\frac{g(y_k)}{g(x_k)} \right) \right\}.$$

This choice of the argument function is consistent with the branch of the complex logarithm used in the numerical experiments.

It can be observed that scheme (4) is obtained by composing the modified Traub–Steffensen’s method twice, incorporating in the second step a weight function H depending on the variable t . On the other hand, the number of functional evaluations

per iteration in scheme (4) is three. Therefore, the method is optimal, since it achieves fourth-order convergence under suitable conditions on the weight function, as stated below. Theorem 1 considers the case $m = 1$.

Theorem 1.

Let $g : D \subset \mathbb{C} \rightarrow \mathbb{C}$ be an analytic function in D surrounding the required root $\alpha \in D$, which is a simple root ($m = 1$) of the equation $g(x) = 0$. If the initial estimate x_0 is sufficiently close to α , then the order of convergence of the iterative scheme defined by (4) is at least four, provided that the weight function H satisfies

$$H(0) = H'(0) = 1, \quad H''(0) = 4, \quad |H'''(0)| < \infty.$$

The error equation is given by

$$e_{k+1} = -\frac{1}{6} \left[6\beta (g'(\alpha))^2 A_1 + (H'''(0) - 30) A_1^2 + 6A_2 \right] A_1 e_k^4 + O(e_k^5),$$

where

$$A_i = \frac{1}{(1+i)!} \frac{g^{(1+i)}(\alpha)}{g'(\alpha)}, \quad i \geq 1,$$

and $e_k = x_k - \alpha$, $k \in \mathbb{N}$, denote the asymptotic error constants and the errors at the k^{th} iteration, respectively.

Theorem 2 considers the general case $m \geq 2$.

Theorem 2.

Let $g : D \subset \mathbb{C} \rightarrow \mathbb{C}$ be an analytic function in D surrounding the required root $\alpha \in D$, which is a solution of multiplicity $m \geq 2$ of the equation $g(x) = 0$. If the initial estimate x_0 is sufficiently close to α , then the order of convergence of the iterative scheme defined by (1) is at least four when the weight function H satisfies

$$H(0) = H'(0) = 1, \quad H''(0) = 4, \quad |H'''(0)| < \infty,$$

and the error equation is given by

$$e_{k+1} = \frac{1}{6m^3} \left[- (H'''(0) - 3(9+m)) B_1^3 - 6mB_1B_2 \right] e_k^4 + O(e_k^5),$$

where

$$B_i = \frac{m!}{(m+i)!} \frac{g^{(m+i)}(\alpha)}{g^{(m)}(\alpha)}, \quad i \geq 1,$$

and $e_k = x_k - \alpha$, $k \in \mathbb{N}$, are the errors in the k^{th} iteration and the asymptotic error constants, respectively.

Remark. It is important to note that the parameter β , used in w_k , appears in the error equation only in the case $m = 1$, but not for $m \geq 2$. For $m \geq 2$, this parameter appears in the coefficient of e_k^4 or higher order terms. However, such terms are unnecessary when establishing fourth order convergence.

Numerical experiments

This section applies the proposed family to different test functions to verify the theoretical results. We select two members of family (4), setting the weight function. In this way, method M1 refers to the polynomial weight function $H_1(t) = 2t^2 + t + 1$ while method M2 is generated with the rational weight function $H_2(t) = \frac{1-t}{1-2t}$. In addition, we take $\beta = \frac{1}{100}$ as the parameter value.

The proposed methods are compared with other multiple-root solvers from the literature. In this sense, we compare with the methods SH [6], BH [7], KS [8] and BM [9].

The multiple-root nonlinear functions under consideration are

- $g_1(x) = x^3 - 5.22x^2 + 9.0825x - 5.2675 = 0$, which has a simple root $\alpha_1 = 1.72$ and a double root $\alpha_2 = 1.75$.
- $g_2(x) = (e^{-x} - 1 + x/5)^3 = 0$, which has a triple root $\alpha \approx 4.96511423174427630369$.
- $g_3(x) = \left(\tan^{-1}(1/x - 1) + e^{x^2} - 3\right)^2 = 0$, which has a double root $\alpha \approx 1.05653$.

The numerical tests were performed in Matlab R2023a on a computer with an Intel Core i7-14700 processor at 2.10 GHz and 16 GB of RAM. The operations use variable-precision arithmetic with 1000 mantissa digits to ensure that the stopping criterion $|x_{k+1} - x_k| + |g(x_k)| < 10^{-50}$ is satisfied without problems of division by zero. A maximum of 50 iterations was also used as a stopping criterion.

The results are presented in Tables 1, 2, y 3. The tables include the initial estimate x_0 , the number of iterations k required to reach the solution, the approximate computational order of convergence (ACOC) [10], the difference between successive iterates $|x_{k+1} - x_k|$, the residual error $|g(x_{k+1})|$, and the CPU time (in seconds) at the last iteration.

Table 1 shows the competitiveness of the proposed methods for the function $g_1(x)$, which contains both a simple root and a double root. The methods M1 and M2 achieve fourth-order convergence and exhibit a performance comparable to the other robust methods considered. In particular, method M2 reaches the solution in fewer iterations than M1 and presents a lower computational time among the proposed schemes.

Table 2 illustrates the numerical behavior of the methods for the function $g_2(x)$, which has a triple root. All methods attain fourth-order convergence with the same number of iterations. However, methods M1 and M2 provide smaller CPU times compared with several of the derivative-based schemes, showing the efficiency of the proposed derivative-free family.

Finally, Table 3 presents the results for the function $g_3(x)$, which has a double root. The proposed methods M1 and M2 preserve fourth-order convergence and provide highly accurate residual errors. In particular, method M2 yields the smallest residual error among the considered methods, while maintaining a competitive computational

cost.

Regarding CPU time, the numerical experiments indicate that the proposed derivative-free methods are computationally efficient. In several cases, methods M1 and M2 require less execution time than some derivative-based methods, which confirms that avoiding derivative evaluations can reduce the overall computational cost while preserving high-order convergence.

Table 1: Methods performance for $g_1(x) = x^3 - 5.22x^2 + 9.0825x - 5.2675$

x_0	Method	k	ACOC	$ x_{k+1} - x_k $	$ g(x_{k+1}) $	CPU Time
2.5	KS	6	4.0	1.681455×10^{-76}	$4.108403 \times 10^{-601}$	0.1600
	BM	7	4.0	$9.179614 \times 10^{-196}$	$3.402403 \times 10^{-1007}$	0.1981
	M1	7	4.0	$5.828891 \times 10^{-169}$	$1.241662 \times 10^{-981}$	0.1835
	M2	6	4.0	1.148375×10^{-70}	$4.375873 \times 10^{-554}$	0.1514
	SH	6	3.9998	5.797116×10^{-52}	$8.663156 \times 10^{-404}$	0.1177
	BH	7	4.0	$2.649541 \times 10^{-178}$	$1.134134 \times 10^{-1007}$	0.1406

Table 2: Methods performance for $g_2(x) = (e^{-x} - 1 + x/5)^3$

x_0	Method	k	ACOC	$ x_{k+1} - x_k $	$ g(x_{k+1}) $	CPU Time
5.5	KS	4	4.0	2.39197×10^{-105}	$3.789736 \times 10^{-1270}$	0.0971
	BM	4	4.0	$1.008215 \times 10^{-104}$	$1.58485 \times 10^{-1262}$	0.1035
	M1	4	4.0	$4.343648 \times 10^{-104}$	$8.599949 \times 10^{-1255}$	0.0888
	M2	4	4.0	$2.427622 \times 10^{-105}$	$4.556872 \times 10^{-1270}$	0.0899
	SH	4	4.0	2.247557×10^{-86}	$6.269454 \times 10^{-1040}$	0.1208
	BH	4	4.0	2.33109×10^{-86}	$9.794113 \times 10^{-1040}$	0.1276

Table 3: Methods performance for $g_3(x) = \left(\tan^{-1}(1/x - 1) + e^{x^2} - 3\right)^2$

x_0	Method	k	ACOC	$ x_{k+1} - x_k $	$ g(x_{k+1}) $	CPU Time
1	KS	6	1.3200	5.899457×10^{-96}	$3.302051 \times 10^{-760}$	0.1613
	BM	5	4.0	5.269261×10^{-81}	$1.133588 \times 10^{-638}$	0.1443
	M1	5	4.0	2.944205×10^{-81}	$1.848931 \times 10^{-640}$	0.1317
	M2	5	4.0	$6.650351 \times 10^{-100}$	$1.495071 \times 10^{-791}$	0.1298
	SH	4	4.0	7.679162×10^{-57}	3.99872×10^{-446}	0.1651
	BH	4	3.9999	5.627046×10^{-52}	$9.615098 \times 10^{-407}$	0.1269

Conclusions

In this work, a family of optimal fourth-order derivative-free iterative schemes has been presented. The construction of the proposed family is based on a modified Traub–Steffensen-type scheme as the first step, while a weight function is incorporated in the second step to enhance the convergence properties.

The numerical implementation shows that the proposed methods M1 and M2 are competitive with respect to other robust methods available in the literature. This performance is evaluated in terms of absolute error between successive iterations, absolute residual error, computational order of convergence, and computational time.

Overall, the results confirm the efficiency and robustness of the proposed family of iterative methods for solving nonlinear equations with multiple roots.

Acknowledgements

Funded with IMM Research Funding Call 2026 – Instituto Universitario de Matemática Multidisciplinar, Universitat Politècnica de València.

References

- [1] R. L. Burden, D. J. Faires y A. M. Burden. *Análisis Numérico*. México, Cengage Learning, 2017.
- [2] E. Schröder. Über unendlich viele Algorithmen zur Auflösung der Gleichungen. *Math. Ann.*, 2:317–365, 1870.
- [3] H. T. Kung y J. F. Traub. Optimal order of one-point and multipoint iteration. *J. Assoc. Comput. Mach.*, 21:643–651, 1974.
- [4] J. F. Traub. *Iterative Methods for the Solution of Equations*. New York, NY, USA, Chelsea Publishing Company, 1982.
- [5] D. Kumar, J. R. Sharma y I. K. Argyros. Optimal one-point iterative function free from derivatives for multiple roots. *Mathematics*, 8:709, 2020.
- [6] J. R. Sharma y R. Sharma. Modified Jarratt method for computing multiple roots. *Applied Mathematics and Computation*, 2217(2):878–881, 2010.
- [7] R. Behl, A. Cordero, S. S. Motsa y J. R. Torregrosa. An optimal fourth-order family of methods for multiple roots and its dynamics. *Numerical Algorithms*, 271:775–796, 2016.
- [8] S. Kumar, D. Kumar, J. R. Sharma, C. Cesarano, P. Agarwal y Y. M. Chu. An Optimal Fourth Order Derivative-Free Numerical Algorithm for Multiple roots. *Symmetry*, 12(1038), 2020.
- [9] R. Behl, S. Bhalla, Á. A. Magreñán y A. Moysi. An Optimal Derivative Free Family of Chebyshev-Halley’s Method for Multiple Zeros. *Mathematics*, 9(546), 2021.
- [10] A. Cordero y J. R. Torregrosa. Variants of Newton’s method using fifth-order quadrature formulas. *Applied Mathematics and Computation*, 190:686–698, 2007.

Framework for computing the Moore–Penrose tensor inverse

J. Pérez–Alemany^{*}, J. C. Azcona–Leiva, and A. Cerdán Martínez

Universitat de València, Av. Vicent Andrés Estellés 19, 46100–Burjassot, Spain

Abstract

Matrices and tensors are fundamental objects in mathematical and numerical modelling, since they are used to define linear transformations, systems of equations, and allow us to perform multivariate data analysis. In this work, we present a novel third-order convergent iterative framework that was originally introduced to compute the inverse of tensors under the Einstein product (EP). We successfully proved that it can be used to compute the more general Moore–Penrose inverse of tensors under the same EP-product. The proposed method, which is a generalisation to the tensor domain of a scalar iterative framework for non-linear systems, outperforms classical methods such as Newton, Chebyshev, and more recently introduced high order Newton-Schulz-type methods. We demonstrate this fact with numerical experiments, where the algorithm requires fewer iterations, EP operations and less CPU time while keeping precision. The results point out that the method is an efficient tool for high-dimensional tensor computations.

Introduction

Matrices and tensors are crucial objects in mathematical and numerical modelling, since they are applied across a wide range of fields such as physics, engineering, finance, machine learning, image processing etc., in order to represent and analyse complex systems.

It is well known that matrices allow us to represent any linear transformation, system of equations, and structured data (e.g., graphs, financial correlations etc.). On the other hand, tensors are a generalisation of matrices, which enable us to extend matrix applications to higher-dimensional contexts. Examples where this extensions are needed could be systems with time-dependent 3D stress fields in mechanics, or multi-layer neural network weights in deep learning (c.f [10, 12, 18, 9]).

Notation 1.

^{*}jopea5@alumni.uv.es

Let us fix $n \in \mathbb{N}$ and define an n -th order tensor as:

$$\mathcal{B} = (b_{i_1 \dots i_n}), \quad 1 \leq i_j \leq P_j \in \mathbb{N}, \quad 1 \leq j \leq n.$$

Therefore, a complex tensor is a multidimensional array that contains $P_1 \times \dots \times P_n$ elements in $\mathbb{C}$. Tensors are often denoted as hyper-matrices, which provides higher-order generalizations of vectors and matrices. We will write the space of all n -th order tensors of this structure by $\mathbb{C}^{P_1 \times \dots \times P_n}$.

Definition 1. ([1])

Let us consider two tensors:

$$\begin{aligned} \mathcal{B} &= (b_{p_1 \dots p_N q_1 \dots q_N}) \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}, \\ \mathcal{C} &= (c_{q_1 \dots q_N k_1 \dots k_M}) \in \mathbb{C}^{Q_1 \times \dots \times Q_N \times K_1 \times \dots \times K_M}, \end{aligned}$$

The EP (Einstein Product) of tensors $\mathcal{B}$ and $\mathcal{C}$, $\mathcal{B} * \mathcal{C} \in \mathbb{C}^{P_1 \times \dots \times P_N \times K_1 \times \dots \times K_M}$, is the tensor defined component-wise as:

$$(\mathcal{B} * \mathcal{C})_{p_1 \dots p_N k_1 \dots k_M} = \sum_{q_N=1}^{Q_N} \dots \sum_{q_1=1}^{Q_1} b_{p_1 \dots p_N q_1 \dots q_N} c_{q_1 \dots q_N k_1 \dots k_M}.$$

We point out that, in the case of $N = M = 1$, the EP simplifies to conventional matrix multiplication.

The computation of *tensor inverses* has been a computationally difficult task due to the complexities of multidimensional data structures. For that reason, *numerical iterative methods* have become an essential tool for efficiently approximating these inverses. In the recent literature, multiple iterative schemes have been developed to compute the Moore-Penrose (MP) inverse of tensors (c.f [6, 7]). For instance, the classical Newton method has been successfully generalised to compute tensor inverses, establishing a powerful framework for high-dimensional computations. The adaptation of Newton's method to the tensor domain is defined as follows [13]:

$$\begin{aligned} \mathcal{X}_{r+1} &= \mathcal{X}_r * (2\mathcal{I} - \mathcal{B} * \mathcal{X}_r) \\ &= \mathcal{X}_r * (\mathcal{I} + \mathcal{R}_j), \\ \mathcal{R}_j &= \mathcal{I} - \mathcal{B} * \mathcal{X}_r, \quad r \geq 0. \end{aligned} \tag{1}$$

Here, $\mathcal{I} \in \mathbb{C}^{P_1 \times \dots \times P_N \times P_1 \times \dots \times P_N}$ is the identity tensor (see Definition 3), and $\mathcal{B} \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$, $\mathcal{X}_r \in \mathbb{C}^{Q_1 \times \dots \times Q_N \times P_1 \times \dots \times P_N}$.

The following theorem establishes the convergence properties of Newton's method:

Theorem 1. ([13])

Let $\mathcal{B} \in \mathbb{C}^{P_1 \times \dots \times P_N \times P_1 \times \dots \times P_N}$ be an invertible tensor (see Definition 4). The iterative method described in Eq.(1) converges to the inverse of $\mathcal{B}$ if the following condition holds:

$$\|\mathcal{I} - \mathcal{B} * \mathcal{X}_0\| \leq q < 1,$$

where q represents a bound on the error norm. Under this condition, the error at the j -th iteration satisfies the inequality:

$$\|\mathcal{X}_j - \mathcal{B}^{-1}\| \leq \frac{\|\mathcal{X}_0\|}{1-q} \|\mathcal{I} - \mathcal{B} * \mathcal{X}_j\| \leq \frac{\|\mathcal{X}_0\|}{1-q} q^{2j}.$$

This result guaranties method's quadratic convergence, which is a well-known result for the matrix and scalar domain. The inequality $\|\mathcal{I} - \mathcal{B} * \mathcal{X}_0\| \leq q < 1$ guarantees both the convergence and efficiency of the method, and conditions for the *initial seed* $\mathcal{X}_0$ that ensures the inequality to hold can be given (c.f [11, 13]).

Another classical method that can be adapted to the tensor domain is the Chebyshev method [14]:

$$\begin{cases} \mathcal{Y}_r = \mathcal{B} * \mathcal{X}_r, \\ \mathcal{X}_{r+1} = \mathcal{X}_r * (3\mathcal{I} - \mathcal{Y}_r * (3\mathcal{I} - \mathcal{Y}_r)), \end{cases} \quad r \geq 0. \quad (2)$$

Recently, Dehdezi and Karimi introduced a fast and efficient Newton-Schulz-type method to find the MP inverse of tensors [13]. This method is an extension of the classical Newton-Schulz iteration, which was originally formulated for matrices, to the tensor domain. It provides accurate and computationally efficient computation of the tensor inverse. Their proposed method is defined as follows:

$$\begin{cases} \mathcal{Y}_r = \mathcal{B} * \mathcal{X}_r, \\ \mathcal{Z}_r = \mathcal{Y}_r * (2\mathcal{I} - \mathcal{Y}_r), \\ \mathcal{X}_{r+1} = \mathcal{X}_r * (2\mathcal{I} - \mathcal{Y}_r) * (3\mathcal{I} - \mathcal{Z}_r * (3\mathcal{I} - \mathcal{Z}_r)), \end{cases} \quad r \geq 0. \quad (3)$$

This self-called fast and efficient iterative approach shows the potential of Newton-Schulz-type methods in advancing tensor-based computations across diverse disciplines.

In this work, we aim to present a novel method for efficiently computing the inverse of tensors, that was introduced in [5]. The main focus of this proposed method is to reduce computational time and achieve greater performance than the methods established above. It was designed to address the challenges caused by high-dimensional tensor operations while maintaining fast convergence and numerical stability.

Preliminary and MP Tensor Inverse

The following section introduces the tensor nomenclature and definitions related with the Einstein-Product (EP) that will be needed in our work.

Definition 2. ([1])

Let $\mathcal{B} = (b_{p_1 \dots p_N q_1 \dots q_M}) \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_M}$ a tensor, then:

$$\|\mathcal{B}\| = \left[\sum_{q_M} \dots \sum_{q_1} \sum_{p_N} \dots \sum_{p_1} |b_{p_1 \dots p_N q_1 \dots q_M}|^2 \right]^{\frac{1}{2}} \geq 0,$$

and $\mathcal{B}^* \in \mathbb{C}^{Q_1 \times \dots \times Q_M \times P_1 \times \dots \times P_N}$, with

$$(\mathcal{B}^*)_{q_1 \dots q_M p_1 \dots p_N} = (\mathcal{B})_{p_1 \dots p_N q_1 \dots q_M}^*$$

are Frobenius norm and conjugate transpose of the tensor $\mathcal{B}$, respectively.

Definition 3. ([1])

A tensor $\mathcal{A} \in \mathbb{C}^{P_1 \times \dots \times P_N \times P_1 \times \dots \times P_N}$ is classified as diagonal when its elements satisfy the condition:

$$a_{p_1 \dots p_N q_1 \dots q_N} = 0, \quad \text{for } p_d \neq q_d, \quad d = 1, \dots, N. \quad (4)$$

Furthermore, a diagonal tensor $\mathcal{I} \in \mathbb{C}^{P_1 \times \dots \times P_N \times P_1 \times \dots \times P_N}$ is designated as an identity tensor if its entries are characterized by:

$$i_{p_1 \dots p_N q_1 \dots q_N} = \prod_{d=1}^N \delta_{p_d q_d}, \quad (5)$$

where, for all $d \in \{1, \dots, N\}$, $\delta_{p_d q_d}$ denotes the Kronecker delta, defined as:

$$\delta_{p_d q_d} = \begin{cases} 1, & p_d = q_d, \\ 0, & p_d \neq q_d. \end{cases}$$

Definition 4. ([1])

Let $\mathcal{A} \in \mathbb{C}^{P_1 \times \dots \times P_N \times P_1 \times \dots \times P_N}$ be a tensor. A tensor $\mathcal{B} \in \mathbb{C}^{P_1 \times \dots \times P_N \times P_1 \times \dots \times P_N}$ is called the inverse of $\mathcal{A}$ with respect to the Einstein-Product (EP) if it satisfies:

$$\mathcal{B} * \mathcal{A} = \mathcal{A} * \mathcal{B} = \mathcal{I},$$

where $\mathcal{I}$ denotes the identity tensor. In this case, we denote $\mathcal{A}^{-1} = \mathcal{B}$.

The following result can be proved straightforwardly:

Proposition 1. ([15])

Let $\mathcal{A} \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$ and $\mathcal{B} \in \mathbb{C}^{Q_1 \times \dots \times Q_N \times K_1 \times \dots \times K_M}$ be tensors. The following properties hold:

$$\begin{aligned} (\mathcal{A} * \mathcal{B})^* &= \mathcal{B}^* * \mathcal{A}^*, \\ \mathcal{I}_N * \mathcal{B} &= \mathcal{B}, \\ \mathcal{B} * \mathcal{I}_M &= \mathcal{B}, \end{aligned}$$

where $\mathcal{I}_N \in \mathbb{C}^{Q_1 \times \dots \times Q_N \times Q_1 \times \dots \times Q_N}$ and $\mathcal{I}_M \in \mathbb{C}^{K_1 \times \dots \times K_M \times K_1 \times \dots \times K_M}$ represent identity tensors of appropriate dimensions.

We now introduce the definition of the Moore–Penrose inverse of tensors in $\mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$ via the Einstein product, which generalizes the Moore–Penrose inverse of matrices.

Definition 5. ([15])

The Moore-Penrose (MP) inverse of a tensor $\mathcal{A} \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$, denoted by $\mathcal{A}^\dagger$, is a tensor $\mathcal{X} \in \mathbb{C}^{Q_1 \times \dots \times Q_N \times P_1 \times \dots \times P_N}$ satisfying the following four conditions:

$$\begin{aligned} \mathcal{A} * \mathcal{X} * \mathcal{A} &= \mathcal{A} \quad (1), & \mathcal{X} * \mathcal{A} * \mathcal{X} &= \mathcal{X} \quad (2), \\ (\mathcal{A} * \mathcal{X})^* &= \mathcal{A} * \mathcal{X} \quad (3), & (\mathcal{X} * \mathcal{A})^* &= \mathcal{X} * \mathcal{A} \quad (4). \end{aligned}$$

In [15] a constructive scheme for constructing the MP inverse is given, which guarantees the existence and uniqueness of the MP inverse in the tensor domain.

Theorem 2. ([15])

The M-P inverse of a tensor $\mathcal{A} \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$ exists and it is unique.

In the literature we can find extensive research on tensor inverses utilizing the EP property. We refer readers to [15, 17, 16].

A Novel Iterative Framework

It is well known that iterative methods for solving non-linear equations are the cornerstone of computational mathematics, encouraging research interest due to their large applicability in scientific and engineering contexts [2, 3, 4]. The objective of this section is to present a novel third-order iterative method for root-finding problems, which is a generalization of the method presented by [5] to functions of a complex variable.

Let $F : D \subseteq \mathbb{C} \rightarrow \mathbb{C}$ be an analytic function in a neighbourhood of a simple root $\alpha \in D$, i.e., satisfying $F(\alpha) = 0$. Given an initial approximation w_0 close to α , the following iterative scheme for locating the root of the equation $F(w) = 0$ is proposed:

$$w_{r+1} = w_r - \left(1 + \frac{1}{2}\gamma + \frac{1}{7}\delta - \frac{1}{21}\zeta + \frac{2}{7}\eta \right) \frac{F(w_r)}{F'(w_r)}, \quad r \geq 0, \quad (6)$$

where the complex-valued parameters, $\gamma, \delta, \zeta, \eta$ are defined as:

$$\begin{aligned} \gamma &= \frac{F''(w_r)F(w_r)}{[F'(w_r)]^2}, \quad \delta = \frac{[F''(w_r)]^2[F(w_r)]^2}{[F'(w_r)]^4}, \\ \zeta &= \frac{F'''(w_r)[F(w_r)]^2}{[F'(w_r)]^3}, \quad \eta = \frac{F^{(4)}(w_r)[F(w_r)]^3}{[F'(w_r)]^4}. \end{aligned}$$

The following theorem establishes the convergence properties of the proposed method, which generalises the classical newton-method for complex functions.

Theorem 3. ([5] Theorem 1 generalisation)

Suppose that $F : D \subseteq \mathbb{C} \rightarrow \mathbb{C}$ is analytic in an open convex set D , and let $\alpha \in D$ be a simple root of the equation $F(w) = 0$, i.e., $F'(\alpha) \neq 0$. Then the iterative method defined in Equation (6) converges to α with a convergence order of at least three. Specifically, the error term satisfies:

$$\epsilon_{r+1} = \frac{5}{7}(2V_2^2 - V_3)(\epsilon_r)^3 + \mathcal{O}((\epsilon_r)^4), \quad (7)$$

where $V_i = \frac{F^{(i)}(\alpha)}{i!F'(\alpha)}$ for $i \geq 2$, and $\epsilon_r = w_r - \alpha$.

Proof

Since F is analytic in the open convex set D , we can expand $F(w_r)$ and its derivatives using Taylor series around α . Let $\epsilon_r = w_r - \alpha$, then:

$$\begin{aligned} F(w_r) &= F'(\alpha) [\epsilon_r + V_2\epsilon_r^2 + V_3\epsilon_r^3 + V_4\epsilon_r^4 + V_5\epsilon_r^5 + V_6\epsilon_r^6 + \mathcal{O}(\epsilon_r^7)], \\ F'(w_r) &= F'(\alpha) [1 + 2V_2\epsilon_r + 3V_3\epsilon_r^2 + 4V_4\epsilon_r^3 + 5V_5\epsilon_r^4 + 6V_6\epsilon_r^5 + \mathcal{O}(\epsilon_r^6)], \\ F''(w_r) &= F'(\alpha) [2V_2 + 6V_3\epsilon_r + 12V_4\epsilon_r^2 + 20V_5\epsilon_r^3 + 30V_6\epsilon_r^4 + \mathcal{O}(\epsilon_r^5)], \\ F'''(w_r) &= F'(\alpha) [6V_3 + 24V_4\epsilon_r + 60V_5\epsilon_r^2 + 120V_6\epsilon_r^3 + \mathcal{O}(\epsilon_r^4)], \\ F^{(4)}(w_r) &= F'(\alpha) [24V_4 + 120V_5\epsilon_r + 360V_6\epsilon_r^2 + \mathcal{O}(\epsilon_r^3)]. \end{aligned}$$

From these expansions, we can compute the complex-valued parameters as:

$$\begin{aligned} \gamma &= 2V_2\epsilon_r + (-6V_2^2 + 6V_3)(\epsilon_r)^2 + \mathcal{O}(\epsilon_r^3), \\ \delta &= 4V_2^2(\epsilon_r)^2 + 24V_2(-V_2^2 + V_3)(\epsilon_r)^3 + \mathcal{O}(\epsilon_r^4), \\ \zeta &= 6V_3(\epsilon_r)^2 + 24(-V_2V_3 + V_4)(\epsilon_r)^3 + \mathcal{O}(\epsilon_r^4), \\ \eta &= 24V_4(\epsilon_r)^3 + 120(-V_2V_4 + V_5)(\epsilon_r)^4 + \mathcal{O}(\epsilon_r^5). \end{aligned}$$

If we substitute these parameters into the iterative scheme Eq.(6) and after algebraic manipulation, we obtain the error propagation:

$$\begin{aligned} w_{r+1} &= w_r - \left(1 + \frac{1}{2}\gamma + \frac{1}{7}\delta - \frac{1}{21}\zeta + \frac{2}{7}\eta\right) \frac{F(w_r)}{F'(w_r)} \\ &= \alpha + \epsilon_r - [1 + V_2\epsilon_r + (2V_2^2 - V_3)(\epsilon_r)^2 + \mathcal{O}(\epsilon_r^3)] \\ &\quad \times [\epsilon_r - V_2\epsilon_r^2 + (2V_2^2 - 2V_3)(\epsilon_r)^3 + \mathcal{O}(\epsilon_r^4)]. \end{aligned}$$

Therefore, after simplification, the dominant error term emerges as:

$$\epsilon_{r+1} = \frac{5}{7}(2V_2^2 - V_3)(\epsilon_r)^3 + \mathcal{O}(\epsilon_r^4),$$

which confirms third-order convergence for complex analytic functions. $\square$

In order to illustrate the connection between scalar and tensor inverses, let us consider the scalar function $w \mapsto F(w) = w^{-1} - b$. If we apply our iterative method from Equation (6) to this function, we obtain the scalar iteration:

$$\begin{aligned} w_{r+1} &= \frac{1}{7}w_r(71 - 219bw_r + 301(bw_r)^2 \\ &\quad - 194(bw_r)^3 + 48(bw_r)^4), \quad r \geq 0. \end{aligned} \tag{8}$$

Motivated by this scalar case and the tensor function $\mathcal{X} \mapsto F(\mathcal{X}) = \mathcal{X}^{-1} - \mathcal{A}$, we now extend our analysis to the tensor domain $\mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$.

Let $\mathcal{A} \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$, the modified iterative method for computing the Moore-Penrose inverse of tensors using the Einstein-Product (EP) based on Eq.(8) is defined as follows:

$$\begin{aligned} \mathcal{X}_{r+1} = & \frac{1}{7} \mathcal{X}_r * \left(71\mathcal{I} - 219\mathcal{A} * \mathcal{X}_r + 301(\mathcal{A} * \mathcal{X}_r)^2 \right. \\ & \left. - 194(\mathcal{A} * \mathcal{X}_r)^3 + 48(\mathcal{A} * \mathcal{X}_r)^4 \right), \quad r \geq 0, \end{aligned} \quad (9)$$

which can be computationally optimized as:

$$\begin{cases} \mathcal{Y}_r = \mathcal{A} * \mathcal{X}_r, \\ \mathcal{Z}_r = \mathcal{Y}_r^2, \\ \mathcal{X}_{r+1} = \frac{1}{7} \mathcal{X}_r * (71\mathcal{I} - 219\mathcal{Y}_r + \mathcal{Z}_r * (301\mathcal{I} - 194\mathcal{Y}_r + 48\mathcal{Z}_r)), \end{cases} \quad r \geq 0. \quad (10)$$

We now present some algebraic properties of the method:

Lemma 1. ([5])

Let $\mathcal{A} \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$. The tensor sequence $\{\mathcal{X}_i\}_{i \geq 0} \subseteq \mathbb{C}^{Q_1 \times \dots \times Q_N \times P_1 \times \dots \times P_N}$ generated by the iterative method Eq.(10), with $\mathcal{X}_0 = \rho \mathcal{A}^*$, $\rho > 0$, satisfies the following algebraic properties:

1. $(\mathcal{X}_i * \mathcal{A})^* = \mathcal{X}_i * \mathcal{A}$,
2. $(\mathcal{A} * \mathcal{X}_i)^* = \mathcal{A} * \mathcal{X}_i$,
3. $\mathcal{X}_i * \mathcal{A} * \mathcal{A}^\dagger = \mathcal{X}_i$,
4. $\mathcal{A}^\dagger * \mathcal{A} * \mathcal{X}_i = \mathcal{X}_i$.

We now introduce a result for tensors that extends the classical result for matrices and provides a sufficient condition for the initial seed of the iterative method to converge to the MP inverse.

Proposition 2. ([13])

Let $\mathcal{A} \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$ with singular values $\sigma_1 > \sigma_2 > \dots > \sigma_r > 0$ and the initial approximation $\mathcal{X}_0 = \alpha \mathcal{A}^*$, where $0 < \alpha < 1/\sigma_1^2$. Then it holds

$$\|\mathcal{A} * (\mathcal{X}_0 - \mathcal{A}^\dagger)\| < 1.$$

Finally, we are in condition to prove our main result:

Theorem 4.

Let $\mathcal{A} \in \mathbb{C}^{P_1 \times \dots \times P_N \times Q_1 \times \dots \times Q_N}$ be a tensor with singular values $\sigma_1 > \sigma_2 > \dots > \sigma_r > 0$. Let the initial approximation be $\mathcal{X}_0 = \alpha \mathcal{A}^*$, where $0 < \alpha < \frac{1}{\sigma_1^2}$. Then, the sequence $\{\mathcal{X}_j\}_{j \geq 0}$ generated by Eq.(10) is convergent to the Moore-Penrose inverse $\mathcal{A}^\dagger$ a third-order rate of accuracy.

Proof

Let $\mathcal{P} = \mathcal{A} * \mathcal{A}^\dagger$ be the orthogonal projector onto the range of $\mathcal{A}$ and let us define the error tensors:

$$\begin{aligned}\mathcal{E}_r &= \mathcal{X}_r - \mathcal{A}^\dagger, \\ \mathcal{U}_r &= \mathcal{Y}_r - \mathcal{P} = \mathcal{A} * \mathcal{X}_r - \mathcal{A} * \mathcal{A}^\dagger \\ &= \mathcal{A} * \mathcal{E}_r, \quad \forall r \geq 0.\end{aligned}$$

Let $r \geq 0$, from Lemma 1 we get $\mathcal{X}_r = \mathcal{A}^\dagger * \mathcal{A} * \mathcal{X}_r = \mathcal{A}^\dagger * \mathcal{Y}_r$. Therefore, the error $\mathcal{E}_r$ satisfies the relation:

$$\begin{aligned}\mathcal{E}_r &= \mathcal{X}_r - \mathcal{A}^\dagger = \mathcal{A}^\dagger * \mathcal{Y}_r - \mathcal{A}^\dagger * \mathcal{A} * \mathcal{A}^\dagger \\ &= \mathcal{A}^\dagger * (\mathcal{Y}_r - \mathcal{P}) = \mathcal{A}^\dagger * \mathcal{U}_r.\end{aligned}$$

Consequently, we have that:

$$\begin{aligned}\|\mathcal{E}_r\| &\leq \|\mathcal{A}^\dagger\| \|\mathcal{U}_r\|, \\ \|\mathcal{U}_r\| &\leq \|\mathcal{A}\| \|\mathcal{E}_r\|.\end{aligned}\tag{11}$$

Let us define the scalar polynomial from Eq.(10)

$$y \mapsto g(y) = \frac{1}{7}y(71 - 219y + y^2(301 - 194y + 48y^2))$$

Because all tensors involved commute, the tensor recurrence for $\mathcal{Y}_r = \mathcal{A} * \mathcal{X}_r$ is exactly the functional relation

$$\mathcal{Y}_{r+1} = g(\mathcal{Y}_r).$$

where $g(\cdot)$ is applied via polynomial functional calculus (valid because $\mathcal{Y}_r$ commutes with itself and $\mathcal{Y}_r * \mathcal{P} = \mathcal{P} * \mathcal{Y}_r = \mathcal{Y}_r$ by Lemma 1). Note that $g(\mathcal{P}) = \mathcal{P}$ since $g(1) = 1$ and $\mathcal{P}^k = (\mathcal{A} * \mathcal{A}^\dagger)^k = \mathcal{A} * \mathcal{A}^\dagger = \mathcal{P}$ for all $k \in \mathbb{N}$ by Lemma 1. Let us define $u := y - 1$, a direct expansion gives the scalar identity:

$$g(1 + u) - 1 = \frac{5}{7}u^3 + \frac{46}{7}u^4 + \frac{48}{7}u^5.$$

Now, since $g(\mathcal{P}) = \mathcal{P}$ and $\mathcal{Y}_r * \mathcal{P} = \mathcal{P} * \mathcal{Y}_r = \mathcal{Y}_r$, we may replace the scalar u by the commuting operator $\mathcal{U}_r = \mathcal{Y}_r - \mathcal{P}$. Hence we have the exact operator identity:

$$\begin{aligned}\mathcal{U}_{r+1} &= \mathcal{Y}_{r+1} - \mathcal{P} = g(\mathcal{Y}_r) - \mathcal{P} = g(\mathcal{P} + \mathcal{U}_r) - \mathcal{P} \\ &= \frac{5}{7}\mathcal{U}_r^3 + \frac{46}{7}\mathcal{U}_r^4 + \frac{48}{7}\mathcal{U}_r^5 \\ &= \left(\frac{5}{7} + \frac{46}{7}\mathcal{U}_r + \frac{48}{7}\mathcal{U}_r^2\right)\mathcal{U}_r^3.\end{aligned}\tag{12}$$

We conclude that:

$$\|\mathcal{U}_{r+1}\| \leq \left(\frac{5}{7} + \frac{46}{7}\|\mathcal{U}_r\| + \frac{48}{7}\|\mathcal{U}_r\|^2\right)\|\mathcal{U}_r\|^3,$$

By Proposition 2 we have that $\|\mathcal{U}_0\| = \|\mathcal{A} * (\mathcal{X}_0 - \mathcal{A}^\dagger)\| < 1$. Then, it follows that $\mathcal{U}_r \rightarrow 0$ as $r \rightarrow +\infty$ with third order accuracy. Furthermore, by Eq.(11)

$$\begin{aligned} \|\mathcal{E}_{r+1}\| &\leq \|\mathcal{A}^\dagger\| \|\mathcal{U}_{r+1}\| \leq \|\mathcal{A}^\dagger\| \left(\frac{5}{7} + \frac{46}{7} \|\mathcal{U}_r\| + \frac{48}{7} \|\mathcal{U}_r\|^2 \right) \|\mathcal{U}_r\|^3 \\ &\leq \|\mathcal{A}^\dagger\| \left(\frac{5}{7} + \frac{46}{7} \|\mathcal{A}\| \|\mathcal{E}_r\| + \frac{48}{7} \|\mathcal{A}\|^2 \|\mathcal{E}_r\|^2 \right) \|\mathcal{A}\|^3 \|\mathcal{E}_r\|^3. \end{aligned}$$

This inequality results $\mathcal{X}_r - \mathcal{A}^\dagger \rightarrow 0$ as $r \rightarrow +\infty$. In other words, Eq.(10) algorithm converges to the Moore-Penrose inverse of $\mathcal{A}$ with a cubic rate of convergence. $\square$

Theorem 5. ([5] Theorem 4 correction)

Let the sequence $\{\mathcal{X}_r\}_{r \geq 0}$ generated by Eq.(10) and suppose that $\|\mathcal{I} - \mathcal{A} * \mathcal{X}_0\| < 1$. For $r \geq 0$ define

$$\Delta_r = \tilde{\mathcal{X}}_r - \mathcal{X}_r,$$

a numerical perturbation that occurs at the r -th exact iterate $\mathcal{X}_r$ of the Eq.(10) algorithm and is so small that we ignore the terms containing $(\Delta_r)^l$ for $l \geq 2$. Then $\|\Delta_{r+1}\| \leq \Gamma(r) \|\Delta_0\|$ for all $r \geq 0$, where:

$$\begin{aligned} \Gamma(r) &= \prod_{j=0}^r [\max\{1, \|\mathcal{E}_j\|^4\} (C + K \|\mathcal{X}_j\| \|\mathcal{A}\|)], \\ C &= \frac{1195}{7}, \quad K = \frac{1343}{7}. \end{aligned}$$

Proof

Let $r \geq 0$ and let us introduce the residual tensors:

$$\begin{aligned} \mathcal{E}_r &= \mathcal{I} - \mathcal{A} * \mathcal{X}_r, \\ \tilde{\mathcal{E}}_r &= \mathcal{I} - \mathcal{A} * \tilde{\mathcal{X}}_r. \end{aligned}$$

We establish the relationship between the exact and approximate residuals:

$$\tilde{\mathcal{E}}_r = \mathcal{E}_r - \mathcal{A} * \Delta_r.$$

The error propagation at iteration $r + 1$ is then given by:

$$\begin{aligned} \Delta_{r+1} &= \tilde{\mathcal{X}}_{r+1} - \mathcal{X}_{r+1} \\ &= \tilde{\mathcal{X}}_r * \left(-\frac{281}{7} \mathcal{I} + \frac{583}{7} \tilde{\mathcal{E}}_r - \frac{281}{7} \tilde{\mathcal{E}}_r^2 + \frac{2}{7} \tilde{\mathcal{E}}_r^3 + \frac{48}{7} \tilde{\mathcal{E}}_r^4 \right) \\ &\quad - \mathcal{X}_r * \left(-\frac{281}{7} \mathcal{I} + \frac{583}{7} \mathcal{E}_r - \frac{281}{7} \mathcal{E}_r^2 + \frac{2}{7} \mathcal{E}_r^3 + \frac{48}{7} \mathcal{E}_r^4 \right). \end{aligned}$$

Substituting $\tilde{\mathcal{X}}_r = \mathcal{X}_r + \Delta_r$ and simplifying yields:

$$\begin{aligned} \Delta_{r+1} &= \Delta_r * \left(-\frac{281}{7} \mathcal{I} + \frac{583}{7} \mathcal{E}_r - \frac{281}{7} \mathcal{E}_r^2 + \frac{2}{7} \mathcal{E}_r^3 + \frac{48}{7} \mathcal{E}_r^4 \right) \\ &\quad + \mathcal{X}_r * \left[\frac{583}{7} (\tilde{\mathcal{E}}_r - \mathcal{E}_r) - \frac{281}{7} (\tilde{\mathcal{E}}_r^2 - \mathcal{E}_r^2) + \frac{2}{7} (\tilde{\mathcal{E}}_r^3 - \mathcal{E}_r^3) + \frac{48}{7} (\tilde{\mathcal{E}}_r^4 - \mathcal{E}_r^4) \right]. \end{aligned}$$

Applying tensor norms and the triangle inequality, we obtain:

$$\begin{aligned} \|\Delta_{r+1}\| &\leq \|\Delta_r\| \left(\frac{281}{7} + \frac{583}{7} \|\tilde{\mathcal{E}}_r\| + \frac{281}{7} \|\tilde{\mathcal{E}}_r\|^2 + \frac{2}{7} \|\tilde{\mathcal{E}}_r\|^3 + \frac{48}{7} \|\tilde{\mathcal{E}}_r\|^4 \right) \\ &\quad + \|\mathcal{X}_r\| \left(\frac{583}{7} \|\tilde{\mathcal{E}}_r - \mathcal{E}_r\| + \frac{281}{7} \|\tilde{\mathcal{E}}_r^2 - \mathcal{E}_r^2\| + \frac{2}{7} \|\tilde{\mathcal{E}}_r^3 - \mathcal{E}_r^3\| + \frac{48}{7} \|\tilde{\mathcal{E}}_r^4 - \mathcal{E}_r^4\| \right). \end{aligned}$$

We now establish bounds for the residual terms. For $k = 1, 2, 3, 4$:

$$\begin{aligned} \|\tilde{\mathcal{E}}_r^k\| &\leq (\|\mathcal{E}_r\| + \|\mathcal{A}\| \|\Delta_r\|)^k = \eta_r^k, \\ \|\tilde{\mathcal{E}}_r^k - \mathcal{E}_r^k\| &= \|(\mathcal{E}_r - \mathcal{A} * \Delta_r)^k - \mathcal{E}_r^k\| \leq \xi_{k,r} \|\mathcal{A}\| \|\Delta_r\|, \end{aligned}$$

where:

$$\begin{aligned} \eta_r &= \|\mathcal{E}_r\| + \mathcal{O}(\|\Delta_r\|), \\ \xi_{k,r} &= \sum_{i=0}^{k-1} \binom{k}{k-1-i} \|\mathcal{E}_r\|^{k-1-i} \|\mathcal{A}\|^i \|\Delta_r\|^i = k \|\mathcal{E}_r\|^{k-1} + \mathcal{O}(\|\Delta_r\|). \end{aligned}$$

If we substitute these bounds into the error estimate we obtain:

$$\begin{aligned} \|\Delta_{r+1}\| &\leq \left[\frac{281}{7} + \frac{583}{7} \eta_r + \frac{281}{7} \eta_r^2 + \frac{2}{7} \eta_r^3 + \frac{48}{7} \eta_r^4 \right. \\ &\quad \left. + \|\mathcal{X}_r\| \|\mathcal{A}\| \left(\frac{583}{7} \xi_{1,r} + \frac{281}{7} \xi_{2,r} + \frac{2}{7} \xi_{3,r} + \frac{48}{7} \xi_{4,r} \right) \right] \|\Delta_r\| \\ &\leq \max\{1, \|\mathcal{E}_r\|^4\} (C + K \|\mathcal{X}_r\| \|\mathcal{A}\|) \|\Delta_r\| + \mathcal{O}(\|\Delta_r\|), \end{aligned}$$

where $C = \frac{281}{7} + \frac{583}{7} + \frac{281}{7} + \frac{2}{7} + \frac{48}{7} = \frac{1195}{7}$ and $K = \frac{583}{7} + \frac{281}{7} 2 + \frac{2}{7} 3 + \frac{48}{7} 4 = \frac{1343}{7}$. Consequently, we have:

$$\|\Delta_{r+1}\| \leq \prod_{j=0}^r [\max\{1, \|\mathcal{E}_j\|^4\} (C + K \|\mathcal{X}_j\| \|\mathcal{A}\|)] \|\Delta_0\| + \mathcal{O}(\|\Delta_r\|),$$

which demonstrates that numerical perturbations remain bounded and the iterative method is asymptotically stable. $\square$

Numerical Experiments

In this section, we present a comprehensive comparative analysis of the computational results obtained from the proposed method and other existing methods that we introduced in Section . All computations were performed on a PC Inter(R) Core(TM) i5-8250U CPU @1.60 GHz (4CPUs), 1800 Mhz, using MATLAB R2024a.

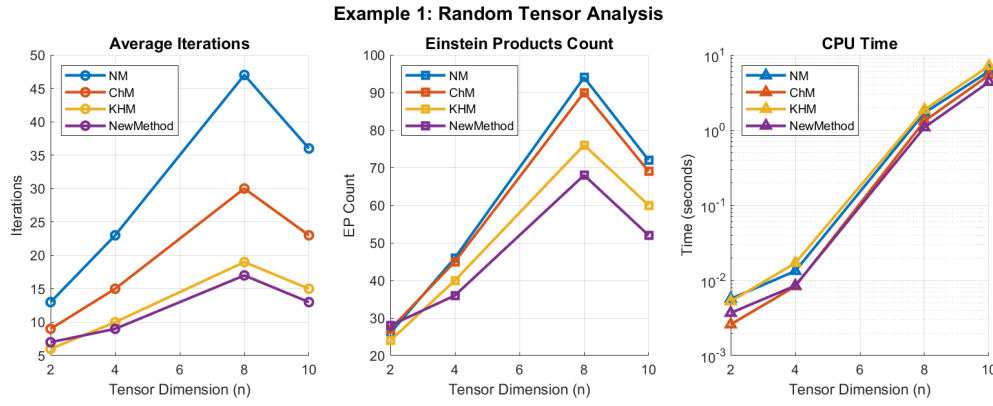


Figure 1: Left to right: average number of iterations, the total count of EPs and CPU time averaged per method in Example 1.

Experimental Setup

We compare the following iterative methods for computing the Moore-Penrose inverse of tensors:

- **NM**: Newton's method (Eq.(1))
- **ChM**: Chebyshev's method (Eq.(2))
- **KHM**: Karimi-Hajarian method (Eq.(3))
- **New Method**: Proposed method (Eq.(10))

Following the approach outlined in [8], we adopt the following stopping criteria for the algorithms:

$$\text{Res} = \frac{\|\mathcal{X}_{r+1} - \mathcal{X}_r\|}{1 + \|\mathcal{X}_r\|} < 10^{-10}. \quad (13)$$

Here, Res represents the *residual* at the termination point, and we will also study the computational time required to reach convergence (CPU-time).

Example 1.

Let us consider random tensors $\mathcal{B} \in \mathbb{R}^{n \times n \times n \times n \times n \times n}$ generated using MATLAB's `tenrand` function:

$$\mathcal{B} = \text{tenrand}(n, n, n, n, n, n) \quad (14)$$

In Figure 1 we present three consecutive runs and present the average number of iterations and Einstein Products (EPs). Additionally, we represent the average CPU time consumption. As it was expected, the computation time increases with tensor dimension, ranging from under 1 second for smaller tensors to significantly longer durations for larger ones. The results demonstrate that the proposed method consistently outperforms existing approaches in terms of convergence speed and computational efficiency for different tensor dimensions and structures.

References

- [1] M. Brazell, N. Li, C. Navasca, and C. Tamon. Solving multilinear systems via tensor inversion. *SIAM Journal on Matrix Analysis and Applications*, 34(2):542–570, 2013.
- [2] Mehdi Dehghan and Masoud Hajarian. Some derivative free quadratic and cubic convergence iterative formulas for solving nonlinear equations. *Computational and Applied Mathematics*, 29, 01 2010.
- [3] R. Erfanifar, K. Sayevand, and H. Esmaeili. On modified two-step iterative method in the fractional sense: some applications in real world phenomena. *International Journal of Computer Mathematics*, 97(10):2109–2141, 2020.
- [4] Raziye Erfanifar. A class of efficient derivative free iterative method with and without memory for solving nonlinear equations. *Computational Mathematics and Computer Modeling with Applications (CMCMA)*, 1(2):20–26, 2022.
- [5] Raziye Erfanifar and Masoud Hajarian. An efficient iterative method to find the moore-penrose inverse of tensors: Applications in image processing and data mining. *Applied Mathematical Modelling*, 151:116426, 2026.
- [6] Raziye Erfanifar, Masoud Hajarian, and Khosro Sayevand. A novel iterative method to find the moore-penrose inverse of a tensor with einstein product. *Numerical Mathematics*, 17(1):37 – 68, 2024. Cited by: 1; All Open Access, Bronze Open Access.
- [7] H. Esmaeili, J. Pirgazi, and J. A. Tenreiro Machado. On the calculation of the Moore–Penrose and Drazin inverses: Application to fractional calculus. *Mathematics*, 9(19):2501, 2021.
- [8] Philip E. Gill, Walter Murray, and Margaret H. Wright. *Numerical Linear Algebra and Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2021.
- [9] Wolfgang Hackbusch. *Tensor spaces and numerical tensor calculus*, volume 42. Springer, 2012.
- [10] Nicholas J Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.
- [11] Nicholas J. Higham. *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2008.
- [12] Piotr Jankowski and Krzysztof Kamil Żur. Flat elliptical matrix crack impact on bending, stresses and stability of beam-like aerospace structures. *Applied Mathematical Modelling*, 142:115963, 2025.
- [13] P. Khosravi Dehdezi and S. Karimi. A rapid and powerful iterative method for computing inverses of sparse tensors with applications. *Journal of Applied Mathematics and Computing*, 68(1):1–19, 2022.

- [14] Hou-Biao Li, Ting-Zhu Huang, Zhen Yun, Li-Tao Li, and Guo Liu. Chebyshev-type methods and preconditioning techniques. *Applied Mathematics and Computation*, 217(12):6015–6026, 2011.
- [15] Lizhu Sun, Baodong Zheng, Changjiang Bu, and Yimin Wei. Moore–penrose inverse of tensors via einstein product. *Linear and Multilinear Algebra*, 64(4):686–698, 2016.
- [16] Lizhu Sun, Baodong Zheng, Yimin Wei, and Changjiang Bu. Generalized inverses of tensors via a general product of tensors. *Frontiers of Mathematics in China*, 13:1–19, 04 2018.
- [17] Yuchao Wang and Yimin Wei. Generalized eigenvalue for even order tensors via einstein product and its applications in multilinear control systems. *Computational and Applied Mathematics*, 41(8):419, 2022.
- [18] Jing-Hua Yang, Xi-Le Zhao, Tian-Hui Ma, Meng Ding, and Ting-Zhu Huang. Tensor train rank minimization with hybrid smoothness regularization for visual data recovery. *Applied Mathematical Modelling*, 81:711–726, 2020.

Frontal surfaces and their singularities

C. Muñoz-Cabello*

Departament de Matemàtiques, Universitat de València, Av. Vicent Andrés Estellés 19, 46100 –Burjassot, España

Abstract

A holomorphic mapping $f: U \subseteq \mathbb{C}^2 \rightarrow \mathbb{C}^3$ is frontal if we can find a holomorphic vector field $\nu: U \rightarrow T\mathbb{C}^3$ such that

$$\|\nu\| \equiv 1; \quad \left\langle \nu, \frac{\partial f}{\partial z} \right\rangle = \left\langle \nu, \frac{\partial f}{\partial w} \right\rangle = 0.$$

An important problem in singularity theory is studying the behaviour of a wave propagating across a homogeneous medium, which can be parametrised as a frontal mapping. In this joint work with J.J. Nuño-Ballesteros and R. Oset Sinha, we study the source double point curve of a frontal surface, which can be described as the set of points $(z, w) \in U$ such that $|f^{-1}(f(z, w))| > 1$. We also define the frontal Milnor number, an important numerical invariant within the context of singularity theory, giving an equation to compute it in terms of the source double point curve.

Introduction

The study of singularities of frontal map germs dates back to the works of Arnold and his colleagues from the University of Moscow between the 70s and 80s [1], in which the concept of Legendrian stability for wave fronts is introduced. A wave front can roughly be defined as the projection of a certain smooth submanifold $\Lambda \hookrightarrow \mathbb{R}^{2n+1}$ via a submersion $\pi: \mathbb{R}^{2n+1} \rightarrow \mathbb{R}^{n+1}$, where Λ and π are subject to a certain compatibility condition with respect to a special structure defined on $\mathbb{R}^{2n+1}$, the contact structure.

Ishikawa extends the notion of Legendrian stability to a broader class of mappings in [3], in which he considers a pair $(\tilde{f}, \pi)$ formed by the same type of submersion as before and a holomorphic mapping $\tilde{f}: U \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^{2n+1}$, again subject to a suitable compatibility condition. Ishikawa also gives a characterisation of Legendrian stability in the language of commutative algebra, what he calls the Arnold-Mather characterisation of Legendrian stability.

Fujimori et al define in [2] the concept of frontal mapping $f: U \subseteq \mathbb{R}^2 \rightarrow \mathbb{R}^3$, showing

*chmuozca@uv.es

that a mapping is frontal if and only if there exists a pair $(\tilde{f}, \pi)$ as described above such that $f = \pi \circ \tilde{f}$. They show that for most points $x \in U$, the germ of a minimal parametrised surface $f: U \rightarrow \mathbb{R}^3$ corresponds to a Legendrian stable pair $(\tilde{f}, \pi)$.

Based on this, Nuño-Ballesteros, Oset Sinha and the author define a theory of frontal stability equivalent to Ishikawa's theory of Legendrian equivalence in [7], with the goal of adapting and extending his results within a framework which minimises the use of contact geometry. Within this context, one problem that emerges is the study of a generic deformation, which can be seen as analyzing how the topology of a moving wave front evolves within a small enough after a time $t > 0$.

Two ways to approach this problem is by computing the multiple point sets of a map germ $f: (\mathbb{C}^n, S) \rightarrow (\mathbb{C}^{n+1}, 0)$ (with $S \subset \mathbb{C}^n$ finite) and its frontal Milnor number. The source k -multiple point set of f can be roughly defined as the set of points $x \in \mathbb{C}^n$ near S such that $f^{-1}(f(x))$ has at least k points, as well as the points x such that the image of f is not smooth around $f(x)$. On the other hand, the frontal Milnor number counts the number of *spheres* emerging within a generic deformation of f – specifically, it is defined as the n -th Betti number of f .

Computing the frontal Milnor number of f usually requires a number of tools from commutative algebra, making it more enticing to instead find formulas involving simpler invariants, such as the number of isolated stable singularities in a stable perturbation and the Milnor number of the source double point set. Thus, the goal of this work is to study the source double point set of f and derive a formula to compute its frontal Milnor number in the case $n = 1$, as well as $n = 2$ when f has corank 1.

Materials

Definition 1

A holomorphic mapping $f: U \subseteq \mathbb{C}^n \rightarrow \mathbb{C}^{n+1}$ with Jacobian matrix J_f has a regular (resp. singular) point in $x \in U$ if the rank of $J_f(x)$ is n (resp. less than n). We will say that f has corank 1 in x if the rank of $J_f(x)$ is $n - 1$.

In the case of mappings with corank 1, we can always choose coordinates in the source and target such that f is given by

$$f(z, w) = (z, p(z, w), q(z, w)); \quad z \in \mathbb{C}^{n-1}, w \in \mathbb{C}$$

in some open neighbourhood of (z, w) .

Definition 2

We say that $f: U \subseteq \mathbb{C}^n \rightarrow \mathbb{C}^{n+1}$, $g: V \subseteq \mathbb{C}^n \rightarrow \mathbb{C}^{n+1}$ share a germ $f: (\mathbb{C}^n, S) \rightarrow (\mathbb{C}^{n+1}, w)$ with $S \subset \mathbb{C}^n$ finite if there is an open $W \subset \mathbb{C}^n$ with $S \subset W$ such that $f(S) = w = g(S)$ and

$$f|_{U \cap W} = g|_{V \cap W}.$$

We say that $f: (\mathbb{C}^n, S) \rightarrow (\mathbb{C}^{n+1}, 0)$ has corank 1 if, for any $x \in S$, f is either regular or has corank 1 at x .

Definition 3

A map germ $f: (\mathbb{C}^2, S) \rightarrow (\mathbb{C}^3, 0)$ with $S \subset \mathbb{C}^2$ finite is frontal if there is a unit holomorphic $\nu: (\mathbb{C}^2, S) \rightarrow T\mathbb{C}^3$ such that

$$\left\langle \nu, \frac{\partial f}{\partial z} \right\rangle = \left\langle \nu, \frac{\partial f}{\partial w} \right\rangle = 0.$$

Consequently, we can define a family of tangent planes $T_x \subset T_x\mathbb{C}^3$ to the image of f such that T_x varies smoothly with respect to x . This allows us to recover and extend results from the differential geometry of surfaces to surfaces with singular points (e.g. [10]).

Theorem 1. [9]

Let $f: (\mathbb{C}^2, S) \rightarrow (\mathbb{C}^3, 0)$ be given by $f(z, w) = (z, p(z, w), q(z, w))$. Then, f is frontal if and only if $p_w|_{q_w} \neq q_w|_{p_w}$.

Definition 4

Given a frontal map germ $f: (\mathbb{C}^2, S) \rightarrow (\mathbb{C}^3, 0)$, an unfolding $F: (\mathbb{C}^d \times \mathbb{C}^2, \{0\} \times S) \rightarrow (\mathbb{C}^d \times \mathbb{C}^3, 0)$ of f is a map germ in the form $F(u, z, w) = (u, f_u(z, w))$ with $f_0 = f$. We will say that f is stable if, given a frontal unfolding F of f , there are diffeomorphic unfoldings Φ, Ψ of the identity such that

$$(\Psi \circ F \circ \Phi^{-1})(u, z, w) = (u, f(z, w)).$$

Figure 1 shows the stable frontal surfaces (besides the embeddings, which are always stable). A parametrisation of each one of these surfaces can be found on Table 1.

Definition 5

If $f: (\mathbb{C}^2, S) \rightarrow (\mathbb{C}^3, 0)$ is a map germ in the form $f(z, w) = (z, p(z, w), q(z, w))$, we define $D^2(f)$ as the set of points $(z, w, w') \in \mathbb{C}^3$ that satisfy the equations

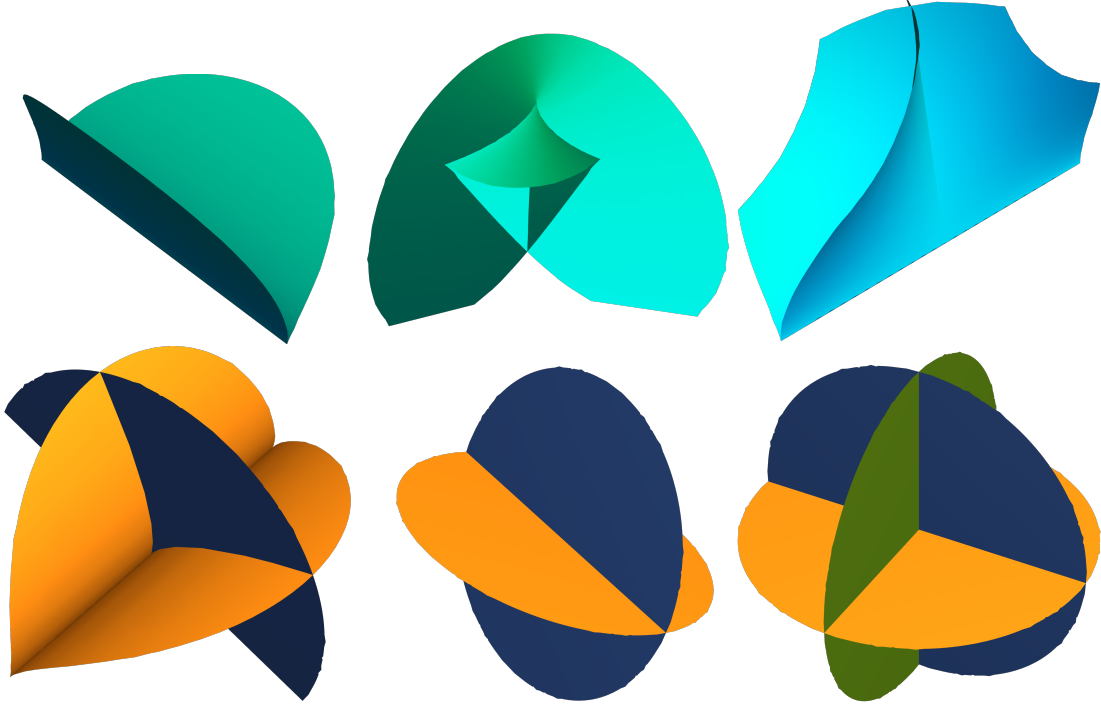
$$\frac{p(z, w) - p(z, w')}{w - w'} = \frac{q(z, w) - q(z, w')}{w - w'} = 0, \quad (1)$$

and call source double point curve to the set $D(f) = \pi(D^2(f))$, where $\pi: \mathbb{C}^3 \rightarrow \mathbb{C}^2$ is the mapping given by $\pi(z, w, w') = (z, w)$.

The equations (1) are well define due to Hilbert's Nullstellensatz (see e.g. [4], Theorem 2.2.6): given a germ of function $h: (\mathbb{C}^2, S) \rightarrow \mathbb{C}$, the set of points in $\mathbb{C}^3$ such that $h(z, w) - h(z, w') = 0$ trivially contains the plane of equation $w = w'$. The Nullstellensatz then implies that $h(z, w) - h(z, w')$ is in the ideal generated by $w - w'$, so there is a germ $H: (\mathbb{C}^3, S') \rightarrow \mathbb{C}$ with $S' = \{(a, b, b) : (a, b) \in S\}$ such that

$$h(z, w) - h(z, w') = H(z, w, w')(w - w') \implies \frac{h(z, w) - h(z, w')}{w - w'} = H(z, w, w').$$

Figure 1: Stable frontal surfaces: cuspidal edge, swallowtail, and folded Whitney umbrella (above); cuspidal double points, transverse double point and transverse triple point (below).



Results

The following results can be found on [8].

Theorem 2.

Let $f: (\mathbb{C}^2, 0) \rightarrow (\mathbb{C}^3, 0)$ be a frontal map germ with corank 1 and generically injective in the form $f(z, w) = (z, p(z, w), q(z, w))$ with $p_w|_{q_w}$. Then, the equation for $D(f)$ is

$$(p_w)^2 \tau = 0$$

for some $\tau: (\mathbb{C}^2, 0) \rightarrow \mathbb{C}$. We then define the source cuspidal edge curve $C(f)$ and the source transverse double point curve $D_+(f)$ as the curves of equation $p_w = 0$ and $\tau = 0$ respectively.

We compute the equations for $C(f)$ and $D_+(f)$ for the stable frontal surfaces on Table 1.

Theorem 3.

Given a corank 1 frontal map germ $f: (\mathbb{C}^2, S) \rightarrow (\mathbb{C}^3, 0)$, there is a stable frontal unfolding $F: (\mathbb{C} \times \mathbb{C}^2, \{0\} \times S) \rightarrow (\mathbb{C} \times \mathbb{C}^3, 0)$. Setting $F(t, z, w) = (t, f_t(z, w))$, we call the family

$$f_t: N_t \rightarrow Z_t$$

with $0 < |t| \ll 1$ a stabilisation of f , and $f_t(N_t)$, a disentanglement of f .

Theorem 4. Frontal Milnor number

Table 1: Double point curve for the stable frontal surfaces.

Symbol	Surface	Parametrisation	C	D_+
A_1	Plane	$(z, w, 0)$	—	—
A_2	Cuspidal edge	(z, w^2, w^3)	w	—
A_1^2	Transverse double point	$(z, w, 0)$ $(0, z', w')$	— —	w z'
$\check{S}_1$	Folded Whitney umbrella	(z, w^2, zw^3)	w	z
A_3	Swallowtail	$(z, w^3 + 3zw, w^4 + 2zw^2)$	$z + w^2$	$3z + w^2$
$A_1 A_2$	Cuspidal double point	$(0, z, w)$ $(z', (w')^2, (w')^3)$	— w'	$z^3 - w^2$ z'
A_1^3	Transverse triple point	$(z, w, 0)$ $(z', 0, w')$ $(0, z'', w'')$	— — —	zw $z'w'$ $z''w''$

If $f: (\mathbb{C}^2, S) \rightarrow (\mathbb{C}^3, 0)$ is a corank 1 frontal map germ and X_t is a disentanglement of f with $0 < |t| \ll 1$,

1. X_t has at most stable singularities;
2. X_t has spheres, but not circles (i.e. $H_1(X_t) = 0$ and $H_2(X_t)$ is torsion-free);
3. the number of spheres (the second Betti number of X_t) does not depend on the choice of F or t .

We denote the number of spheres as $\mu_{\mathcal{F}}(f)$, and call it the frontal Milnor number of f .

Theorem 5. Frontal Marar-Mond formulas

Let $f: (\mathbb{C}^2, 0) \rightarrow (\mathbb{C}^3, 0)$ be a corank 1 frontal map germ in the form

$$f(z, w) = (z, p(z, w), q(z, w))$$

with $q_w = \mu p_w$ for some $\mu: (\mathbb{C}^2, 0) \rightarrow \mathbb{C}$. If f is generically injective and the set

$$\{(z, w) \in \mathbb{C}^2 : p_w(z, w) = \mu_w(z, w) = 0\}$$

is isolated near 0,

$$\begin{aligned} \mu(f(C(f)), 0) &= 2|A_3| + \mu(C(f), 0); \\ 2\mu(f(D_+(f)), 0) &= 2|A_1 A_2| + 2|A_1^3| + \mu(D_+(f), 0) - |\check{S}_1| - |A_3| + 1. \end{aligned}$$

Here, $\mu(\cdot, 0)$ denotes the Milnor number at 0, and $|X|$ denotes the number of singularities of type X (see Table 1) in a disentanglement of f .

Theorem 6.

Let $f: (\mathbb{C}, 0) \rightarrow (\mathbb{C}^2, 0)$ be a frontal map germ with $\kappa(f)$ singular points in a stabilisation. Then,

$$\mu_{\mathcal{F}}(f) - \mu_I(f) = \kappa(f),$$

where μ_I denotes the image Milnor number of f .

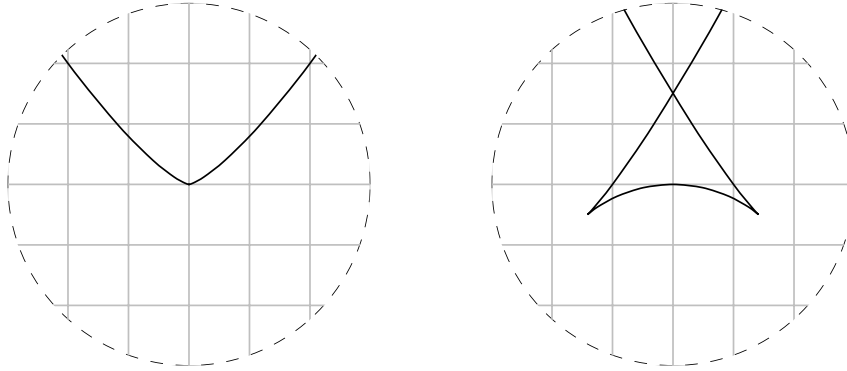


Figure 2: Real image of the frontal map germ $f: (\mathbb{C}, 0) \rightarrow (\mathbb{C}^2, 0)$ given by $f(x) = (x^3, x^4)$ (left) along its stabilisation for $t = 1/2$ (right). The point $(0, 0)$ splits into two singular points (thus $\kappa(f) = 2$) and a node (transverse double point).

Theorem 7.

Let $f: (\mathbb{C}^2, 0) \rightarrow (\mathbb{C}^3, 0)$ be a corank 1 frontal map germ. If f has at most stable singularities near the origin,

$$\mu_{\mathcal{F}}(f) = \mu(f(D_+), 0) + |A_3| + |\check{S}_1| - 3|A_1^3| - 2|A_2A_1| - 1.$$

Conclusions

A more general expression to compute the frontal Milnor number of a frontal map germ f is given on [6], but it requires computing a stable unfolding beforehand. The identities given on Theorems 6 and 7, while only valid for corank 1 map germs, only require knowing f itself, as the number of isolated stable singularities can be counted using the expressions given on [8]. These expressions also give an immediate geometric interpretation of the frontal Milnor number: from the point of view of algebraic geometry, the stable frontal singularities and spheres emerging on a disentanglement X_t are regarded as topological information that emerges from within the origin, much like how ripples emerge from the point where a pebble hits the surface of a puddle or pop corn pops from a corn kernel (see also Figure 2). In this way, Theorems 6 and 7 quantify the contributions made by $D(f)$ and the stable singularities to $\mu_{\mathcal{F}}(f)$.

Acknowledgements

This work is partially supported by Grant PID2021-124577NB-I00 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe".

References

- [1] V. I. Arnol' d, S. M. Gusein-Zade and A. Varchenko, *Singularities of differentiable maps. Vol. I*, traducido del ruso por Ian Porteous y Mark Reynolds, Monographs in Mathematics, 82, Birkhäuser Boston, Boston, MA, 1985.
- [2] S. Fujimori et al., Singularities of maximal surfaces, *Math. Z.* **259** (2008), no. 4, 827–848.
- [3] G. Ishikawa, Infinitesimal deformations and stability of singular Legendre submanifolds, *Asian J. Math.* **9** (2005), no. 1, 133–166.
- [4] T. de Jong and G. Pfister, *Local analytic geometry*, Advanced Lectures in Mathematics, Friedr. Vieweg, Braunschweig, 2000
- [5] D. Mond, Vanishing cycles for analytic maps, in *Singularity theory and its applications, Part I (Coventry, 1988/1989)*, 221–234, Lecture Notes in Math., 1462, Springer, Berlin
- [6] C. Muñoz-Cabello, J. J. Nuño Ballesteros and R. Oset Sinha, A proof of the Mond conjecture for wave fronts, *Bull. Braz. Math. Soc. (N.S.)* **56** (2025), no. 2, Paper No. 21, 25 pp.; MR4881689
- [7] C. Muñoz-Cabello, J. J. Nuño Ballesteros and R. Oset Sinha, Deformations of corank 1 frontals, *Proc. Roy. Soc. Edinburgh Sect. A* **154** (2024), no. 4, 1204–1234.
- [8] C. Muñoz-Cabello, J. J. Nuño Ballesteros and R. Oset Sinha, Singularities of frontal surfaces, *Hokkaido Math. J.* **53** (2024), no. 1, 175–208; MR4705701
- [9] J. J. Nuño Ballesteros, Unfolding plane curves with cusps and nodes, *Proc. Roy. Soc. Edinburgh Sect. A* **145** (2015), no. 1, 161–174.
- [10] R. Oset Sinha and K. Saji, The axial curvature for corank 1 singular surfaces, *Tohoku Math. J. (2)* **74** (2022), no. 3, 365–388.

Iterative methods for approximating the Moore-Penrose inverse of complex matrices

Juan Carlos Azcona Leiva*, Aarón Cerdán Martínez, and Joan Pérez Alemany

Universitat de València, Av. Vicent Andrés Estellés 19, 46100-Burjassot, Spain

Abstract

The Moore-Penrose inverse has numerous applications in various areas of mathematics, such as total least squares problems, solving linear systems and statistical regression analysis. Over the last decade, new iterative schemes for computing the Moore-Penrose inverse have been developed. This work examines a general family of methods of arbitrary order, which encompasses several methods reported in the literature. In particular, we focus on a subclass given by a natural order p and a real parameter $\beta \in [0, 1]$. Additionally, we analyse an efficient ninth-order method that requires seven matrix multiplications per iteration. Inspired by the latter, we propose a twelfth-order method that uses only eight matrix multiplications per iteration. Finally, we provide numerical experiments to test the performance of the studied schemes.

Introduction

The Moore-Penrose inverse represents one of the most widely used generalized inverses, with applications ranging from least-squares problems to statistical regression. In this work, we focus on iterative methods for computing this inverse for complex matrices. We denote the conjugate transpose of a complex matrix $A \in \mathbb{C}^{m \times n}$ by A^* .

Definition 1

Let $A \in \mathbb{C}^{m \times n}$. Its **Moore-Penrose inverse**, denoted by $A^\dagger$, is defined as the unique matrix $A^\dagger \in \mathbb{C}^{n \times m}$ that satisfies the following equations:

$$AA^\dagger A = A, \quad A^\dagger AA^\dagger = A^\dagger, \quad (AA^\dagger)^* = AA^\dagger, \quad (A^\dagger A)^* = A^\dagger A. \quad (1)$$

Now, we present a proposition given in [9] that affirms the Moore-Penrose inverse is well-defined.

*azlei@alumni.uv.es

Proposition 1

Let $A \in \mathbb{C}^{m \times n}$ be a complex matrix. Then $A^\dagger$ exists, and it is unique.

Moreover, [12] states that if $\text{rank}(A) = \min\{n, m\}$, we can express $A^\dagger$ as

$$A^\dagger = \begin{cases} (A^*A)^{-1}A^*, & \text{if } m > n \\ A^{-1}, & \text{if } m = n \\ A^*(AA^*)^{-1}, & \text{if } m < n \end{cases}$$

Note that if $m \neq n$, the Moore-Penrose inverse coincides with the minimum square solution, whereas if A is non-singular, $A^\dagger = A^{-1}$. In consequence, all proved methods for the Moore-Penrose inverse are valid for the usual inverse. Now we introduce the following notation.

Notation 1

Let $k, p \in \mathbb{N}$. We will denote the binomial coefficient “ p choose k ” as C_k^p , i.e.,

$$C_k^p := \binom{p}{k} = \frac{p!}{k!(p-k)!}, \quad 0 \leq k \leq p.$$

For commodity, whether $k > p$, we define $C_k^p := 0$.

Some of the most well-known schemes for computing the Moore-Penrose inverse can be expressed as

$$X_{k+1} = X_k \sum_{i=1}^p \lambda_i (AX_k)^{i-1}, \quad k \geq 0 \quad (2)$$

where A is the given matrix, X_0 the initial approximation and $\lambda_i \in \mathbb{R}$ are parameters such that $\sum_{i=1}^p \lambda_i = 1$. For example, if $p = 2$ and $\lambda_1 = 2, \lambda_2 = -1$, then Eq.(2) is the classic Newton-Schulz method (see [11]), while if $\lambda_i = (-1)^{i-1} C_i^p$, for $1 \leq i \leq p$, then Eq.(2) is the hyperpower method of order p (see [3]). For this reason, it would be worthwhile to study whether the scheme Eq.(2) converges to $A^\dagger$ and with which order it does.

A General Class for Computing the Moore-Penrose Inverse: The Cordero-Soto-Quiros-Torregrosa Family

In May 2021, Cordero, Soto-Quiros and Torregrosa proposed in [5] a new class of iterative schemes for computing the Moore-Penrose inverse similar to Eq.(2). To present it, we first need to define a polynomial matrix.

Definition 2

Let $p > 0$ be a natural and $B \in \mathbb{C}^{m \times m}$ a square complex matrix. We define $G_p(B)$ as

$$G_p(B) := \sum_{j=1}^p (-1)^{j-1} C_j^p B^{j-1}.$$

Now, let $p > 0$ be a natural, $\alpha_i \in \mathbb{R}$, $1 \leq i \leq p$ real parameters, $A \in \mathbb{C}^{m \times n}$ a complex matrix and X_0 a given approximation of $A^\dagger$. The family of schemes considered in [5] is:

$$X_{k+1} = X_k \sum_{i=1}^p \alpha_i G_i(AX_k), \quad k \geq 0 \quad (3)$$

The authors found sufficient conditions for the convergence of this scheme.

Theorem 1.

Let $A \in \mathbb{C}^{m \times n}$, $p \in \mathbb{N}$ and suppose for each $1 \leq i \leq p$, $\alpha_i \in [0, 1]$ such that $\sum_{i=1}^p \alpha_i = 1$ and $\alpha_p \neq 0$. Consider a constant $\alpha < \frac{1}{\sigma_1^2}$ where σ_1 denotes the largest singular value of A . Then, if $X_0 := \alpha A^*$, the sequence Eq.(3) converges to the Moore-Penrose inverse of A with order $q := \min_{1 \leq i \leq p} \{i : \alpha_i \neq 0\}$.

Notice that for every matrix $X \in \mathbb{C}^{m \times m}$ and every $n \in \mathbb{N}$, the sets $V := \{X^0, X^1, \dots, X^{n-1}\}$ and $W := \{G_1(X), G_2(X), \dots, G_n(X)\}$ are two bases of the polynomial space $\mathbb{C}_{n-1}[X]$. It is clear from the definition of $G_p(X)$ that the change-of-basis matrix from V to W is:

$$(\mathcal{M}_{V \rightarrow W})_{j,i} = \begin{cases} (-1)^{j-1} C_j^i, & \text{if } j \leq i, \\ 0, & \text{if } j > i. \end{cases}$$

Also, one can verify that for every $n \in \mathbb{N}$, $(\mathcal{M}_{V \rightarrow W})^2 = I$. Consequently,

$$\mathcal{M}_{W \rightarrow V} = (\mathcal{M}_{V \rightarrow W})^{-1} = \mathcal{M}_{V \rightarrow W}.$$

Therefore, we have an explicit form for converting between the scheme Eq.(2) and the one in Eq.(3), given that the former is more simplified, while we know sufficient convergence conditions of the latter. For simplicity, we will write $\mathcal{M}$ instead of $\mathcal{M}_{V \rightarrow W}$. In fact, the matrix $\mathcal{M}$ has the following property:

Proposition 2

Let $p \in \mathbb{N}$ be a natural and $\lambda = (\lambda_i)_i \in \mathbb{R}^p$ a column vector of real parameters. Set $\alpha := \mathcal{M}\lambda$, i.e.,

$$X_k \sum_{i=1}^p \lambda_i (AX_k)^{i-1} = X_k \sum_{i=1}^p \alpha_i G_i(AX_k), \quad k \geq 0.$$

Then, $\sum_{i=1}^p \lambda_i = \sum_{i=1}^p \alpha_i$.

Proof

By definition of $\mathcal{M}$, $\alpha_i = \sum_{j=i}^p (-1)^{i-1} C_i^j \lambda_j$. Thus,

$$\begin{aligned}
 \sum_{i=1}^p \alpha_i &= \sum_{i=1}^p \left(\sum_{j=i}^p (-1)^{i-1} C_i^j \lambda_j \right) \\
 &= \sum_{j=1}^p \lambda_j \left(\sum_{i=1}^j (-1)^{i-1} C_i^j \right) \\
 &= - \sum_{j=1}^p \lambda_j \left(\sum_{i=1}^j (-1)^i C_i^j \right) \\
 &= - \sum_{j=1}^p \lambda_j \left(\sum_{i=0}^j (-1)^i C_i^j - 1 \right) \\
 &= - \sum_{j=1}^p \lambda_j ((1-1)^j - 1) \\
 &= \sum_{j=1}^p \lambda_j,
 \end{aligned}$$

as claimed. □

As a consequence, any scheme of the form of Eq.(2) such that $\sum_{i=1}^p \lambda_i = 1$ will converge to $A^\dagger$ given that $\alpha_i := \sum_{j=i}^p (-1)^{i-1} C_i^j \lambda_j$ satisfies $\alpha_i \in [0, 1]$, $1 \leq i \leq p$ and $\alpha_p \neq 0$. By using this relation, it is easy to verify which methods of the literature are a particular case of the Cordero-Soto-Quiros-Torregrosa family (Eq. 3). The authors themselves include a dozen examples in [5]. Some of them are:

1. For any $p > 1$, the parameters $\alpha_i = 0$, $1 \leq i \leq p-1$ and $\alpha_p = 1$ provides the hyperpower method of order p . In particular, for $p = 2$ we have the Newton-Schulz method and for $p = 3$ the Chebyshev one (see [8]).
2. For $\alpha_1 = \alpha_2 = \alpha_3 = 0$ and $\alpha_4 = \alpha_5 = \frac{1}{2}$ we get the method proposed by Toutounian and Soleymani [15]:

$$X_{k+1} = \frac{1}{2} X_k (9I - AX_k (16I - AX_k (14I - AX_k (6I - AX_k))), \quad k \geq 0.$$

3. Let $\alpha \in (0, 1]$ be a fixed parameter. If $\alpha_1 = 1 - \alpha$, $\alpha_2 = 0$ and $\alpha_3 = \alpha$, then we get the method proposed by Srivastava and Gupta in [14]:

$$X_{k+1} = X_k (I + \alpha (2I - 3AX_k + (AX_k)^2)), \quad k \geq 0.$$

Additionally, we mention two well-known methods that can also be expressed in the form of Eq.(3). The first one is the Homeier method [8]:

$$X_{k+1} = X_k \left(I + \frac{1}{2} (I - AX_k) (I + (2I - AX_k)^2) \right), \quad k \geq 0.$$

If we rewrite it as

$$X_{k+1} = \frac{1}{2}X_k(7I - 9AX_k + 5(AX_k)^2 - (AX_k)^3), \quad k \geq 0,$$

by using the change-of-basis matrix $\mathcal{M}$, the desired parameters are $\alpha_1 = \alpha_2 = 0$ and $\alpha_3 = \alpha_4 = \frac{1}{2}$. The second one is the Mid-Point scheme, also found in [8]:

$$X_{k+1} = \left(I + \frac{1}{4}(I - AX_k)(3I - AX_k)^2 \right) X_k, \quad k \geq 0.$$

Similarly, by expressing it as

$$X_{k+1} = \frac{1}{4}X_k(13I - 15AX_k + 7(AX_k)^2 - (AX_k)^3), \quad k \geq 0,$$

we obtain $\alpha_1 = \alpha_2 = 0$, $\alpha_3 = \frac{3}{4}$ and $\alpha_4 = \frac{1}{4}$. Observe that in both cases $\min\{i : \alpha_i \neq 0\} = 3$, which is consistent with the literature that affirms that both are third-order methods.

An Interesting Subclass of the General Family

In July 2023, Kansal et al. [6] studied the following family of methods:

$$X_{k+1} = X_k((3 + \beta)I + (-3 - 3\beta)AX_k + (1 + 3\beta)(AX_k)^2 - \beta(AX_k)^3), \quad (4)$$

where $\beta \in [0, 1]$ is a real variable. One can verify that, by taking certain values of β , we recover some methods mentioned in the previous section. If we take $\beta = 0$, we obtain the Chebyshev method; for $\beta = \frac{1}{2}$, the Homeier method; for $\beta = \frac{1}{4}$, the Midpoint method; and for $\beta = 1$, the hyperpower method of order 4.

The authors were able to prove in [6] that, for an initial estimate $X_0 = \alpha A^*$ with a suitable $\alpha \in \mathbb{R}$, the sequence of approximations generated by Eq.(4) converges to $A^\dagger$ for $\beta \in [0, 1]$. Moreover, they found that this convergence is of order 3 for $\beta \in [0, 1)$ and of order 4 for $\beta = 1$.

Moreover, in [4], Cordero et al. analysed the following family of fourth-order iterative schemes for solving nonlinear scalar equations $f(x) = 0$:

$$\begin{aligned} y_k &= x_k - \frac{f(x_k)}{f'(x_k)}, \\ z_k &= y_k - \beta \frac{f(y_k)}{f[x_k, y_k]}, \\ x_{k+1} &= z_k - \frac{f(z_k)}{f[y_k, z_k]}, \quad k \geq 0, \end{aligned} \quad (5)$$

where $\beta \in \mathbb{R}$ is a parameter and $f[x_k, y_k] := \frac{f(y_k) - f(x_k)}{y_k - x_k}$ is the divided difference. By substituting the matrix equation $F(X) := X^{-1} - A$ in Eq.(5) they arrived to the family defined as:

$$\begin{aligned} X_{k+1} &= X_k((4 + \beta)I + AX_k(-2(3 + 2\beta)I + AX_k(2(2 + 3\beta)I \\ &\quad + AX_k(-(1 + 4\beta)I + \beta AX_k))), \quad k \geq 0. \end{aligned} \quad (6)$$

Cordero et al. proved in [4] that for a suitable X_0 , the sequence $\{X_k\}$ obtained by Eq.(6) converges to A^{-1} with convergence order 4 if $\beta \in [0, 1)$ and convergence order if $\beta = 1$. They also showed the fifth-order convergence to $A^\dagger$ in the case of $\beta = 1$. Note that the method Eq.(6) can be expressed as the form of Eq.(2):

$$X_{k+1} = X_k((4 + \beta)I + (-6 - 4\beta)AX_k + (4 + 6\beta)(AX_k)^2 + (-1 - \beta)(AX_k)^3 + \beta(AX_k)^4), \quad k \geq 0. \quad (7)$$

One can readily check that in both families Eq.(4) and Eq.(7), the coefficients of the powers $(AX_k)^i$ sum to 1, independently of the value of β .

Thus, inspired by these works, we propose a generalisation of those schemes. That is, we are interested in working with a family of the form Eq.(2) where $p \in \mathbb{N}$, $p \geq 3$ is fixed and $\sum_{i=1}^p \lambda_i = 1$. As in [6] and [4], we introduce a variable $\beta \in \mathbb{R}$. Our proposal is choosing $\alpha_i = 0$ for $1 \leq i \leq p-2$, $\alpha_{p-1} = 1 - \beta$ and $\alpha_p = \beta$ in the Cordero-Soto-Quiros-Torregrosa family. Then, it is clear that by defining $\alpha := (\alpha_i)_i \in \mathbb{R}^p$ and $\lambda := \mathcal{M}\alpha$, that

$$\lambda_i = (-1)^{i-1}C_i^{p-1}(1 - \beta) + (-1)^{i-1}C_i^p\beta, \quad i = 1, \dots, p,$$

or, equivalently

$$\lambda_i = (-1)^{i-1}(C_i^{p-1} + C_{i-1}^{p-1}\beta), \quad i = 1, \dots, p. \quad (8)$$

By Proposition 2, we have $\sum_{i=1}^p \lambda_i = 1$ as we wanted.

This choice of the λ_i 's has a clear motivation. On one hand, it can be verified that for $p = 4$, they coincide with the coefficients given in [6] and, for $p = 5$, they coincide with the coefficients presented in [4]. Thus, if $p = 4$, it was shown that for $\beta = 0$, we obtained the Chebyshev method, i.e., the hyperpower method of order $p-1 = 3$, while for $\beta = 1$, we obtained the hyperpower method of order $p = 4$. The same relationship holds with $p = 5$. Therefore, we aim to extend this relation for any $p \in \mathbb{N}$. Now, note that as we mentioned before, if we substitute the parameters $\alpha_i = 0$ for $1 \leq i \leq p-1$ and $\alpha_p = 1$ in Eq.(3), we obtain the hyperpower method of order p . Because of the election of our α_i 's, the connection is trivial for any $p \geq 2$.

Furthermore, we have all the conditions of the Theorem 1, therefore, we can state the next result that confirms the convergence of our proposed families and provides their convergence order:

Theorem 2.

Let $A \in \mathbb{C}^{m \times n}$, let $p \in \mathbb{N}$, $p \geq 3$ be fixed, and let $X_0 = \alpha A^$ be an initial estimate for a constant $\alpha \in \mathbb{R}$ such that $\|(X_0 - A^\dagger)A\| < 1$. Then the sequence of approximations generated by*

$$X_{k+1} = X_k \sum_{i=1}^p (-1)^{i-1} (C_i^{p-1} + C_{i-1}^{p-1}\beta) (AX_k)^{i-1}, \quad k \geq 0, \quad (9)$$

i.e., the scheme Eq.(2) with the parameters from Eq.(8), converges to $A^\dagger$ for $\beta \in [0, 1]$. Moreover, it has convergence order of at least $p - 1$ for $\beta \in [0, 1)$ and convergence order of at least p for $\beta = 1$.

Note that we include $\alpha_p := \beta = 0$, given that in this case, we can omit the last parameter and obtain the hyperpower method of order $p - 1$.

Although the proof of the latter theorem is not required, we present it to illustrate a common procedure for verifying the validity of an iterative method and deducing its convergence order. Thus, let us state a preliminary lemma, analogous to the one presented in [6].

Lemma 1

Let $X_0 = \alpha A^*$ be the initial approximation, with $\alpha \in \mathbb{R}$. The sequence $\{X_k\}$ generated by the scheme Eq.(9) satisfies for all $k = 0, 1, 2, \dots$

$$\mathbf{MP1:} \quad A^\dagger A X_k = X_k,$$

$$\mathbf{MP2:} \quad X_k A A^\dagger = X_k,$$

$$\mathbf{MP3:} \quad (X_k A)^* = X_k A,$$

$$\mathbf{MP4:} \quad (A X_k)^* = A X_k.$$

Proof

We will prove this by induction on k . For $k = 0$, since $X_0 = \alpha A^*$ it is clear that the 4 properties hold:

$$A^\dagger A X_0 = \alpha A^\dagger A A^* = \alpha (A^\dagger A)^* A^* = \alpha (A A^\dagger A)^* = \alpha A^* = X_0,$$

$$X_0 A A^\dagger = \alpha A^* A A^\dagger = \alpha A^* (A A^\dagger)^* = \alpha (A A^\dagger A)^* = \alpha A^* = X_0,$$

$$(X_0 A)^* = (\alpha A^* A)^* = \alpha A^* (A^*)^* = X_0 A,$$

$$(A X_0)^* = (\alpha A A^*)^* = (A^*)^* (\alpha A^*) = A X_0.$$

Now suppose these properties are true for some $k \geq 0$. We prove the case $k + 1$. Firstly, we show **MP1**:

$$A^\dagger A X_{k+1} = A^\dagger A X_k \left(\sum_{i=1}^p \lambda_i (A X_k)^{i-1} \right) = X_k \left(\sum_{i=1}^p \lambda_i (A X_k)^{i-1} \right) = X_{k+1}.$$

Similarly, we can prove **MP2**, although we must note that for any $n \in \mathbb{N}$, $X_k (A X_k)^n = (X_k A)^n X_k$. Therefore, we can rewrite Eq.(9) as

$$X_{k+1} = \left(\sum_{i=1}^p \lambda_i (X_k A)^{i-1} \right) X_k$$

and thus:

$$X_{k+1} A A^\dagger = \left(\sum_{i=1}^p \lambda_i (X_k A)^{i-1} \right) X_k A A^\dagger = \left(\sum_{i=1}^p \lambda_i (X_k A)^{i-1} \right) X_k = X_{k+1}.$$

Now, we prove **MP3**:

$$\begin{aligned}
 (X_{k+1}A)^* &= \left(X_k \left(\sum_{i=1}^p \lambda_i (AX_k)^{i-1} \right) A \right)^* \\
 &= \left(\sum_{i=1}^p \lambda_i (X_k A)^i \right)^* \\
 &= \sum_{i=1}^p \lambda_i ((X_k A)^i)^* \\
 &= \sum_{i=1}^p \lambda_i (X_k A)^i \\
 &= X_k \sum_{i=1}^p \lambda_i (AX_k)^{i-1} A \\
 &= X_{k+1}A.
 \end{aligned}$$

Analogously, one can prove **MP4**. □

Having seen this lemma, we will prove the theorem that guarantees convergence and provides the order. For that purpose, let us first recall the definition of order of convergence:

Definition 3

Let $\{X_k\}$ be a sequence of matrices that converges to X , with $X_k \neq X$ for every $k \geq 0$. We say that $\{X_k\}$ converges to X with **order** p if there exists $\lambda > 0$ such that

$$\lim_{k \rightarrow +\infty} \frac{\|X_{k+1} - X\|}{\|X_k - X\|^p} = \lambda.$$

Therefore, we provide the proof of Theorem 2:

Proof

To prove convergence it suffices to show that

$$\lim_{k \rightarrow +\infty} \|X_{k+1} - A^\dagger\| = 0.$$

For this purpose, use the properties of $A^\dagger$ and those provided by the previous lemma:

$$\|X_{k+1} - A^\dagger\| = \|X_{k+1}AA^\dagger - A^\dagger AA^\dagger\| \leq \|X_{k+1}A - A^\dagger A\| \|A^\dagger\|. \quad (10)$$

Thus, if we compute:

$$\begin{aligned}
X_{k+1}A - A^\dagger A &= \left(\sum_{i=1}^p \lambda_i (X_k A)^{i-1} \right) X_k A - A^\dagger A \\
&= \sum_{i=1}^p (-1)^{i-1} (C_i^{p-1} + C_{i-1}^{p-1} \beta) (X_k A)^i - A^\dagger A \\
&= \sum_{i=1}^{p-1} (-1)^{i-1} C_i^{p-1} (X_k A)^i + \beta \sum_{i=1}^p (-1)^{i-1} C_{i-1}^{p-1} (X_k A)^i - A^\dagger A.
\end{aligned}$$

Now, the Pascal identity is well known, which states that $C_k^n = C_{k-1}^{n-1} + C_k^{n-1}$ for any $n, k \in \mathbb{N}, n > k$; from this it follows that $C_{i-1}^{p-1} = C_i^p - C_i^{p-1}$. Note that by defining $C_p^{p-1} := 0$, the result holds for $i = p$. Thus:

$$\begin{aligned}
X_{k+1}A - A^\dagger A &= \\
&= \sum_{i=1}^{p-1} (-1)^{i-1} C_i^{p-1} (X_k A)^i + \beta \sum_{i=1}^p (-1)^{i-1} (C_i^p - C_i^{p-1}) (X_k A)^i - A^\dagger A \\
&= (1 - \beta) \sum_{i=1}^{p-1} (-1)^{i-1} C_i^{p-1} (X_k A)^i + \beta \sum_{i=1}^p (-1)^{i-1} C_i^p (X_k A)^i - A^\dagger A.
\end{aligned}$$

Now, check that $(X_k A)^n (A^\dagger A)^m = (X_k A)^n$ for any $n, m \in \mathbb{N}, n \geq 1$. Fix $n \in \mathbb{N}$ and proceed by induction on m . For $m = 0$ it is trivial. By the induction hypothesis, assume it holds for some $m \geq 0$. Then:

$$\begin{aligned}
(X_k A)^n (A^\dagger A)^{m+1} &= (X_k A)^n (A^\dagger A)^m A^\dagger A \\
&= (X_k A)^n A^\dagger A \\
&= (X_k A)^{n-1} X_k A A^\dagger A \\
&= (X_k A)^{n-1} X_k A \\
&= (X_k A)^n.
\end{aligned}$$

Hence, the result follows. Moreover, note that

$$-A^\dagger A = -(1 - \beta)A^\dagger A - \beta A^\dagger A.$$

Combining these facts, we get:

$$\begin{aligned}
X_{k+1}A - A^\dagger A &= \\
&= (1 - \beta) \sum_{i=0}^{p-1} (-1)^{i-1} C_i^{p-1} (X_k A)^i A^\dagger A + \beta \sum_{i=0}^p (-1)^{i-1} C_i^p (X_k A)^i A^\dagger A \\
&= (1 - \beta) \sum_{j=0}^{p-1} (-1)^{p-j} C_{p-1-j}^{p-1} (X_k A)^{p-1-j} A^\dagger A + \beta \sum_{j=0}^p (-1)^{p-1-j} C_{p-j}^p (X_k A)^{p-j} A^\dagger A.
\end{aligned}$$

Now, on the one hand, recall that for any $n, k \in \mathbb{N}, n \geq k$, $C_{n-k}^n = C_k^n$. On the other hand, recall the binomial expansion:

$$(x - y)^n = \sum_{k=0}^n (-1)^k C_k^n x^{n-k} y^k.$$

Moreover, for any $j \in \mathbb{N}$ we have $(-1)^{-j} = (-1)^j$. Therefore, it follows that:

$$\begin{aligned} X_{k+1}A - A^\dagger A &= (-1)^p(1 - \beta) \sum_{j=0}^{p-1} (-1)^j C_j^{p-1} (X_k A)^{p-1-j} (A^\dagger A)^j \\ &\quad - (-1)^p \beta \sum_{j=0}^p (-1)^j C_j^p (X_k A)^{p-j} (A^\dagger A)^j \\ &= (-1)^p \left((1 - \beta)(X_k A - A^\dagger A)^{p-1} - \beta(X_k A - A^\dagger A)^p \right). \end{aligned}$$

In summary, we have shown that if we define $F_k := X_k A - A^\dagger A$ for $k \geq 0$, then

$$F_{k+1} = (-1)^p((1 - \beta)F_k^{p-1} - \beta F_k^p), \quad k \geq 0. \quad (11)$$

Now we will show that $\lim_{k \rightarrow +\infty} \|F_k\| = 0$ for $\beta \in [0, 1]$. For this, proceed as in [1]. Define the sequence $s_k := \|F_k\|$. By induction, we will prove that $s_k < 1$. For $k = 0$ this is true by the hypothesis of the theorem. Suppose it holds for $k \geq 0$, then, by applying the triangle inequality to Eq.(11):

$$s_{k+1} \leq (1 - \beta)s_k^{p-1} + \beta s_k^p \leq (1 - \beta)s_k + \beta s_k = s_k < 1. \quad (12)$$

Hence, we have shown not only that $s_k < 1$ for every $k \geq 0$ but also that $\{s_k\}$ is a monotonically decreasing sequence. As it is bounded below by 0, it converges to a limit s with $0 \leq s < 1$.

Suppose, for contradiction, that $s \neq 0$. Then from Eq.(12) it follows that:

$$s \leq (1 - \beta)s^{p-1} + \beta s^p$$

Then, if we divide by $s \neq 0$ and use that $s^n \leq s$ for any $n \geq 1$:

$$1 \leq (1 - \beta)s^{p-2} + \beta s^{p-1} \leq (1 - \beta)s + \beta s = s.$$

But $s < 1$, yielding a contradiction. Thus, $s = 0$, i.e.

$$s = \lim_{k \rightarrow +\infty} \|F_k\| = \lim_{k \rightarrow +\infty} \|X_k A - A^\dagger A\| = 0.$$

Therefore, applying the Squeeze Theorem to relation Eq.(10), it follows that

$$\lim_{k \rightarrow +\infty} \|X_k - A^\dagger\| = 0,$$

i.e., the family of methods Eq.(9) is convergent for $\beta \in [0, 1]$.

Finally, it remains to determine the order of convergence. For this, define $E_k := X_k - A^\dagger$ for each $k \geq 0$. Note that $E_k A = X_k A - A^\dagger A = F_k$. Then, if $\beta \in [0, 1)$, since $s_k = \|F_k\| < 1$:

$$\|F_{k+1}\| \leq (1 - \beta)\|F_k\|^{p-1} + \beta\|F_k\|^p \leq (1 - \beta)\|F_k\|^{p-1} + \beta\|F_k\|^{p-1} = \|F_k\|^{p-1}.$$

Therefore,

$$\|E_{k+1}A\| = \|F_{k+1}\| \leq \|F_k\|^{p-1} = \|E_k A\|^{p-1} \leq \|E_k\|^{p-1} \|A\|^{p-1}.$$

Hence:

$$\begin{aligned} \lim_{k \rightarrow +\infty} \frac{\|X_{k+1} - A^\dagger\|}{\|X_k - A^\dagger\|^{p-1}} &\leq \lim_{k \rightarrow +\infty} \frac{\|X_{k+1}A - A^\dagger A\| \|A^\dagger\|}{\|E_k\|^{p-1}} \\ &= \lim_{k \rightarrow +\infty} \frac{\|E_{k+1}A\| \|A^\dagger\|}{\|E_k\|^{p-1}} \\ &\leq \lim_{k \rightarrow +\infty} \frac{\|E_k\|^{p-1} \|A\|^{p-1} \|A^\dagger\|}{\|E_k\|^{p-1}} \\ &= \|A\|^{p-1} \|A^\dagger\|, \end{aligned}$$

so if $\beta \in [0, 1)$ the convergence order is at least $p - 1$.

On the other hand, if $\beta = 1$:

$$\|F_{k+1}\| \leq (1 - \beta)\|F_k\|^{p-1} + \beta\|F_k\|^p = \|F_k\|^p,$$

so

$$\|E_{k+1}A\| = \|F_{k+1}\| \leq \|F_k\|^p = \|E_k A\|^p \leq \|E_k\|^p \|A\|^p.$$

Similarly, it follows that the convergence order is p . □

Corollary 1

The previous theorem can be extended to $p = 2$ provided we do not consider the case $\beta = 0$.

Proof

All the development carried out in the previous proof remains valid up to the point where the limit $s = \lim_{k \rightarrow +\infty} s_k$ is introduced. Again, we know that $0 \leq s < 1$. Then from Eq.(12):

$$s \leq (1 - \beta)s + \beta s^2 \implies 0 \leq \beta(s^2 - s).$$

In this case, we assume $\beta \neq 0$, and therefore either $s = 0$ or $s \geq 1$. Since $s < 1$, necessarily $s = 0$ and convergence follows.

The discussion about the order of convergence is the same as in the previous theorem. The only omission is $\beta = 0$, because the method does not converge in that case. □

Observe that with this theorem, we have proven the convergence and the order for the Newton–Schulz method (take $p = 2$ and $\beta = 1$) and for the hyperpower method of order n (take $p = n$ and $\beta = 1$).

In the theorem we studied, we set $X_0 = \alpha A^*$, where we assumed that α is a real number such that $\|\alpha A^* A - A^\dagger A\| < 1$. Therefore, it is of interest to determine such an α , and a sufficient condition can be found in [1]:

Proposition 3

Let $\sigma_1(A), \dots, \sigma_r(A) > 0$ be the non-zero singular values of the matrix $A \in \mathbb{C}^{m \times n}$, and let $\lambda_1(A^* A), \dots, \lambda_r(A^* A) > 0$ be the non-zero eigenvalues of $A^* A$, i.e., $(\sigma_i(A))^2 = \lambda_i(A^* A)$ for $1 \leq i \leq \min\{m, n\}$. Then, if

$$\max_{1 \leq i \leq r} |1 - \alpha \lambda_i(A^* A)| < 1,$$

it follows that $\|\alpha A^* A - A^\dagger A\| < 1$.

Also, in [3] Chen and Wang deduce this result, which is the one used by [5]:

Proposition 4

Let $\sigma_1 > 0$ be the largest single value of a complex matrix $A \in \mathbb{C}^{m \times n}$. If $\alpha < \frac{1}{\sigma_1^2}$ is a constant, there exists a matrix norm $\|\cdot\|$ such that $\|\alpha A^* A - A^\dagger A\| < 1$.

A High-order Efficient Scheme for Computing the Moore-Penrose Inverse

In general, a higher-order method implies a higher computational cost per iteration. For example, we observe the hyperpower method of convergence order p , which can be expressed as:

$$X_k \sum_{i=1}^p (-1)^{i-1} C_i^p (AX_k)^{i-1}, \quad k \geq 0.$$

One can appreciate that each iteration requires exactly p matrix multiplications. In relation to this phenomenon, Soleymani defined in [13] an index to compare the computational complexity of any Schulz-type method:

Definition 4

Let ρ be the order of convergence of the method and let κ be the number of matrix-by-matrix multiplications in floating point arithmetics with machine precision. The **Inverse-finder informational efficiency index**, expressed by $II EI$, is the ratio

$$II EI := \frac{\rho}{\kappa}$$

Thus, the objective is finding methods where $II EI \geq 1$, i.e, $\kappa \leq \rho$.

Furthermore, Soleymani presents in [13] the following scheme to compute the Moore-Penrose inverse:

$$X_{k+1} = -\frac{1}{4} X_k W_n (-4I + K_n W_n), \quad k \geq 0, \quad (13)$$

where $K_n = AX_k$ and

$$W_n = 9I + K_n(-16I + K_n(14I + K_n(-6I + K_n))).$$

This is an eighth-order method that requires seven matrix-by-matrix multiplications for iteration. Thus, has a greater *II EI* than the hyperpower method of order eight.

A similar efficient scheme is described in [10] by Kafei Razavi et al.:

$$X_{k+1} = -\frac{1}{4}X_k W_n(-4I + K_n W_n), \quad k \geq 0, \quad (14)$$

where $K_n = AX_k$ and

$$W_n = -11I + K_n(25I + K_n(-30I + K_n(20I + K_n(-7I + K_n)))).$$

This method involves one more matrix multiplication at each iteration than the latter, but increases the order of convergence by two units.

In 2021, Kaur, Kansal and Kumar [7] also proposed a ninth-order scheme:

$$\begin{aligned} X_{k+1} = & \frac{1}{8}X_k(84I - 402AX_k + 1159(AX_k)^2 - 2241(AX_k)^3 + 3060(AX_k)^4 \\ & - 3024(AX_k)^5 + 2178(AX_k)^6 - 1134(AX_k)^7 + 416(AX_k)^8 \\ & - 102(AX_k)^9 + 15(AX_k)^{10} - (AX_k)^{11}), \quad k \geq 0. \end{aligned}$$

One can appreciate that it involves 12 matrix multiplications per iteration, which adds to the computational cost. Nevertheless, the authors found the following factorisation:

$$\begin{cases} V_k = AX_k, \\ T_k = -7I + V_k(9I + V_k(-5I + V_k)), \\ S_k = V_k T_k, \\ X_{k+1} = -\frac{1}{8}X_k T_k(12I + S_k(6I + S_k)). \end{cases} \quad (15)$$

By doing this change, they minimised the computational workload, as the scheme now presents only seven matrix multiplications per iteration. This also implies that this method has greater *II EI* than Eq.(13). One can appreciate that, in fact, Eq.(15) belongs to the Cordero-Soto-Quiros-Torregrosa family. The choice of parameters is $p = 12$, $\alpha_i = 0$ for $1 \leq i \leq 8$, $\alpha_9 = \alpha_{12} = \frac{1}{8}$ and $\alpha_{10} = \alpha_{11} = \frac{3}{8}$.

Inspired by this work, we provide a high-order scheme with larger *II EI* than Eq.(13) (*II EI* = $\frac{8}{7} \approx 1.14$), Eq.(14) (*II EI* = $\frac{10}{8} = 1.25$) and Eq.(15) (*II EI* = $\frac{9}{7} \approx 1.29$). We can benefit from the work of [5], by choosing the suitable α 's. Consequently, no proof is required for either the validity of the method or its order of convergence.

Our election is $p = 16$, $\alpha_i = 0$ for $1 \leq i \leq 11$, $\alpha_{12} = \alpha_{16} = \frac{1}{16}$, $\alpha_{13} = \alpha_{15} = \frac{4}{16}$ and $\alpha_{14} = \frac{6}{16}$, that provides a method of convergence $q := \{i : \alpha_i \neq 0\} = 12$. If we use the

change-of-basis matrix $\mathcal{M}$, we get:

$$\begin{aligned} X_{k+1} = & \frac{1}{16} X_k (224I - 1464AX_k + 5928(AX_k)^2 - 16641(AX_k)^3 \\ & + 34332(AX_k)^4 - 53834(AX_k)^5 + 65428(AX_k)^6 \\ & - 62271(AX_k)^7 + 46552(AX_k)^8 - 27236(AX_k)^9 \\ & + 12336(AX_k)^{10} - 4239(AX_k)^{11} + 1068(AX_k)^{12} \\ & - 186(AX_k)^{13} + 20(AX_k)^{14} - (AX_k)^{15}), \quad k \geq 0. \end{aligned}$$

Although the scalars can seem arbitrary, they are the coefficients of the polynomial

$$p(x) := -\frac{1}{x}((1-x)^{12}(2-x)^4 - 2^4).$$

We appreciate that this method involves 16 matrix multiplications at each iteration, which is less favourable than the hyperpower method of order 12. Despite that, by proceeding similarly than [7], we can factorise the latter expression as it follows:

$$\begin{cases} V_k = AX_k, \\ T_k = -7I + V_k(9I + V_k(-5I + V_k)), \\ S_k = V_k T_k, \\ X_{k+1} = -\frac{1}{16} X_k T_k (32I + S_k(24I + S_k(8I + S_k))). \end{cases} \quad (16)$$

Thus, our proposed scheme requires only 8 matrix multiplications per iteration, which implies that $III E I = \frac{12}{8} = 1.5$, representing a significant improvement over the previously discussed schemes.

Finally, we present the following conjecture as a generalisation of high-order efficient methods:

Conjecture 1

Let $p, d \in \mathbb{N}$ such that $d \mid p$. Then, the scheme of the Cordero-Soto-Quiros-Torregrosa family where $\alpha_i = 0$ for $1 \leq i < p$ and $\alpha_i = \frac{1}{2^d} C_{i-p}^d$ for $p \leq i \leq p + d$, has order p and can be expressed with $\frac{p}{d} + d + 1$ matrix multiplications per iteration.

One can check that if $p = 9, d = 3$, it is the Kaur-Kansal-Kumar method and if $p = 12, d = 4$, it is our proposed method.

Numerical experiments

In this section, we incorporate our proposed method Eq.(16) in a numerical comparison presented in [7]. All computations were performed on a CPU Intel(R) Core(TM) i5-1035G1 CPU 1.00GHz and using MATLAB R2025b.

We compare the following Schulz-type methods for approximating the Moore-Penrose inverse:

- **KRM:** Kafeai Razavi et al. method Eq.(14).

- **KM:** Kaur-Kansal-Kumar method Eq.(15).
- **HP8M:** Hyperpower method of order nine.
- **PM:** Proposed method Eq.(16).

For this evaluation (example 5 in [7]), we consider twenty random square matrices of order $n = 100$. The entries are complex numbers z where $\text{Re}z \in [-2, 2]$ and $\text{Im}z \in [-1, 1]$. The initial approximation, suggested by Ben-Israel and Greville [2], is $X_0 = \alpha A^*$, where $0 < \alpha < \frac{2}{\|A\|_2^2}$. In this case, we choose $\alpha = \frac{1}{\|A\|_2^2}$. Meanwhile, the stop criterion is:

$$\frac{\|X_{k+1} - X_k\|_\infty}{1 + \|X_{k+1}\|_\infty} < 10^{-8}.$$

Additionally, we repeat the same experiments in the case where the matrices are rectangular, with a dimension of 100×120 . The results are presented in the following table:

Table 1: Comparison of different methods for random matrices

Methods	<i>HPM</i>	<i>KM</i>	<i>KRM</i>	<i>PM</i>
Order of random matrices is 100×100				
Average Time	0.0168	0.0127	0.0139	0.0126
Average Iterations	8.00	7.60	7.40	7.00
Order of random matrices is 100×120				
Average Time	0.0092	0.0088	0.0091	0.0091
Average Iterations	5.00	5.00	5.00	5.00

Remarked in bold the superior result.

As expected, our proposed method requires the fewest number of iterations in both cases. Moreover, our proposed method computed the Moore-Penrose inverse for squared matrices in less time, showing its efficiency. Also, we can appreciate that the hyperpower method, which has the least *IIEI*, is less efficient than the others. An interesting observation can be made between *KRM* and *PM*. Both require 8 matrix multiplications, but the latter has a more favourable outcome, as its convergence order is two units greater. Finally, in the case of matrices of order 100×120 , the difference between the four methods is minimal. This phenomenon can be explained by the fact that we are comparing high-order schemes of similar order.

Acknowledgements

The authors would like to thank the Universitat de València and the Universitat Politècnica de València for organizing the XIII Congress of the Master in Mathematical Research and Doctorate in Mathematics.

References

- [1] Zainab Abu-Iram. Iterative methods for moore-penrose inverse, 2018.
- [2] Adi Ben-Israel and Thomas NE Greville. *Generalized inverses: theory and applications*, volume 15. Springer Science & Business Media, 2003.
- [3] Haibin Chen and Yiju Wang. A family of higher-order convergent iterative methods for computing the moore-penrose inverse. *Applied Mathematics and Computation*, 218(8):4012–4016, 2011.
- [4] Alicia Cordero, Elaine Segura, Juan R Torregrosa, and María P Vassileva. New iterative procedures for approximating different types of inverse matrix: Convergence and stability. *Mathematical Methods in the Applied Sciences*, 48(8):8806–8817, 2025.
- [5] Alicia Cordero, Pablo Soto-Quiros, and Juan R Torregrosa. A general class of arbitrary order iterative methods for computing generalized inverses. *Applied Mathematics and Computation*, 409:126381, 2021.
- [6] Munish Kansal, Manpreet Kaur, Litika Rani, and Lorentz Jäntschi. A cubic class of iterative procedures for finding the generalized inverses. *Mathematics*, 11(13):3031, 2023.
- [7] Manpreet Kaur, Munish Kansal, and Sanjeev Kumar. An efficient matrix iterative method for computing moore-penrose inverse. *Mediterranean Journal of Mathematics*, 18(2):42, 2021.
- [8] Hou-Biao Li, Ting-Zhu Huang, Yong Zhang, Xing-Ping Liu, and Tong-Xiang Gu. Chebyshev-type methods and preconditioning techniques. *Applied Mathematics and Computation*, 218(2):260–270, 2011.
- [9] Roger Penrose. A generalized inverse for matrices. In *Mathematical proceedings of the Cambridge philosophical society*, volume 51, pages 406–413. Cambridge University Press, 1955.
- [10] M Kafaei Razavi, Asghar Kerayechian, Mortaza Gachpazan, and Stanford Shateyi. A new iterative method for finding approximate inverses of complex matrices. In *Abstract and Applied Analysis*, volume 2014, page 563787. Wiley Online Library, 2014.
- [11] Günther Schulz. Iterative berechnung der reziproken matrix. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 13(1):57–59, 1933.
- [12] Salman Sheikhi and Hamid Esmaeili. A new iterative method to find the moore-penrose inverse. 2023.
- [13] Fazlollah Soleymani. A fast convergent iterative solver for approximate inverse of matrices. *Numerical Linear Algebra with Applications*, 21(3):439–452, 2014.

- [14] Shwetabh Srivastava and Dharmendra Kumar Gupta. A third order iterative method for a. *International Journal of Computing Science and Mathematics*, 4(2):140–151, 2013.
- [15] Faezeh Toutounian and Fazlollah Soleymani. An iterative method for computing the approximate inverse of a square matrix and the moore–penrose inverse of a non-square matrix. *Applied Mathematics and Computation*, 224:671–680, 2013.

Iterative Methods for the Drazin Inverse of a Matrix with Real and Complex Spectrum

Aarón Cerdán Martínez*, Juan Carlos Azcona Leiva, and Joan Pérez Alemany

Universitat de València, Av. Vicent Andrés Estellés 19, 46100-Burjassot, Spain

Summary

The Drazin inverse has many applications in several mathematical areas such as: singular differential and difference equations, Markov chains, cryptography, iterative methods, numerical analysis... Although Drazin inverse is only defined for square matrices, it has important spectral properties that the Moore-Penrose pseudoinverse does not always satisfy. Its wide range of applications and the theoretical properties mentioned motivate its study.

This paper is divided in two parts: first, we will examine the construction process of the Drazin inverse through its spectral properties, this will help us to understand further results of the work. Then, we will study some methods to compute Drazin inverse for matrices with real spectrum. These methods will be generalized in order to compute the Drazin inverse of matrices with spectrum in the right open half-plane of the complex plane. Finally, we will show some numerical examples to illustrate the algorithms developed.

Introduction

The Drazin inverse plays an important role in a variety of mathematical applications, including singular differential and difference equations, Markov chains, cryptography, iterative methods, and numerical analysis, among others. Beyond its applications, the Drazin inverse is of particular interest because, despite being defined only for square matrices, it enjoys spectral properties that are not generally satisfied by the Moore-Penrose pseudoinverse.

The first part of this paper is devoted to the spectral characterization of the Drazin inverse. Following the approach of [3], we develop the theoretical framework that underlies the rest of the work. In the second part, we focus on computational aspects.

*acermar4@posgrado.upv.es

We review the methods proposed in [2] for matrices with real spectrum and the extensions introduced in [1] for matrices whose spectrum lies in the open right half-plane of $\mathbb{C}$. The paper concludes with several numerical examples illustrating the performance of these algorithms.

Construction of Drazin inverse

All the matrices studied in this section have coefficients in $\mathbb{C}$ unless otherwise indicated. We begin giving some notation and refreshing some definitions. Given a matrix $A \in \mathbb{C}^{n \times n}$, we denote its image space as $R(A) := \{Az : z \in \mathbb{C}^{n \times 1}\}$. Its null space is $N(A) := \{z \in \mathbb{C}^{n \times 1} : Az = 0\}$.

The index k of A is defined as the smallest natural such that: $\text{rank } A^k = \text{rank } A^{k+1}$. We denote the index as $\text{Ind}(A)$.

Motivation for the construction of the Drazin Inverse

These are the Moore-Penrose equations studied before:

$$\begin{aligned} AA^\dagger A &= A \quad (1) & A^\dagger AA^\dagger &= A^\dagger \quad (2) \\ (AA^\dagger)^* &= AA^\dagger \quad (3) & (A^\dagger A)^* &= A^\dagger A \quad (4). \end{aligned}$$

We can ask ourselves if only choosing some of these equations allows us to build other pseudoinverses (under some conditions). These possibilities can be also explored from the study of the eigenvalues of the matrices. This is the reason why we will only study square matrices (they are the only ones with eigenvalues).

We will see that for square matrices there is a strong connection between some systems of equations that define pseudoinverses and the spectral properties of those pseudoinverses (which are very intuitive).

In this section we will only work with square matrices (note that this allows us to multiply a matrix by itself, what enables us to generalize some of the Moore-Penrose equations). Let X be the pseudoinverse of A , we will also add new equations about commutativity of the product AX (note that the equations (5^k) and (6^k) only have dimensional sense for square matrices, that is why they do not appear in Moore-Penrose formulas):

$$\begin{aligned} A^k X A &= A^k, \quad (1^k) & AX &= XA, \quad (5) \\ A^k X &= XA^k, \quad (5^k) & AX^k &= X^k A. \quad (6^k) \end{aligned}$$

Definition 1

Given $A \in \mathbb{C}^{n \times n}$ and λ one of its eigenvalues, x is called an eigenvector of A (associated to λ) of degree p (or λ -vector of degree p) if it satisfies:

$$(A - \lambda I)^p x = 0$$

Note that if $p = 1$, we obtain the traditional definition of eigenvector.

One of the properties of non-singular matrices (those whose determinant is not zero, i.e., its eigenvalues are not null) is that

Proposition 1

Given A a non-singular matrix,

x is an eigenvector of A (associated to λ) of degree p if x is an eigenvector of A^{-1} (associated to λ^{-1}) of degree p .

The objective is to study pseudoinverses for singular matrices that satisfy the previous property, i.e., we want to generalize the proposition 1 to singular matrices (under some conditions). That is the reason why we will study group pseudoinverses.

Spectral inverses: S-inverses

In order to introduce group pseudoinverses we will study how to compute pseudoinverses of diagonalizable matrices (not necessary invertible, they can admit the null eigenvalue).

In this case, the eigenvalues of the pseudoinverse are defined as follows:

$$\lambda^\dagger = \begin{cases} \lambda^{-1} & \text{si } \lambda \neq 0 \\ 0 & \text{si } \lambda = 0 \end{cases}$$

We are searching for a matrix X whose eigenvalues are $\lambda_1^\dagger, \lambda_2^\dagger, \dots, \lambda_n^\dagger$.

As A has n linearly independent vectors, the matrix P of eigenvectors is non-singular and satisfies:

$$AP = PJ,$$

where $J = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$.

Because of the way that we have defined the eigenvalues of X , it must satisfy an analogous condition:

$$XP = PJ^\dagger,$$

where $J = \text{diag}(\lambda_1^\dagger, \lambda_2^\dagger, \dots, \lambda_n^\dagger)$.

$A = P J P^{-1}$ and $X = P J^\dagger P^{-1}$. As J and $J^\dagger$ are diagonalizable they commute, from where we get that X satisfies the equations (1), (2) and (5). Remember that we have been studying diagonalizable matrices, the next step will be to analyze the case of non-diagonalizable matrices. We sum up this ideas by giving a definition:

Definition 2

Given a square matrix A , X (the pseudoinverse of A) is an S -inverse if, for any $\lambda \in \mathbb{C}$ and x complex vector:

x is a λ -vector of A of degree p if and only if x is a $\lambda^\dagger$ -vector of X of degree p .

Group pseudoinverse

Definition 3

The group inverse of a square matrix A is the pseudoinverse that satisfies (1), (2) and (5) (note that we do not ask A to be diagonalizable), it is denoted by $A^\#$. If $A^\#$ exists, it is unique (the proof can be found in [3]).

The name of group inverse comes from the fact that $\{A^n : n \in \mathbb{N}\} \cup \{(A^\#)^n : n \in \mathbb{N}\} \cup \{AA^\#\}$ is an abelian group.

Now, we give a characterization of the existence of the group inverse:

Theorem 1.

Given a square matrix $A \in \mathbb{C}^{n \times n}$, $A^\#$ exists, if and only if $R(A)$ y $N(A)$ are complementary spaces, i.e., $\mathbb{C}^{n \times 1} = R(A) \oplus N(A)$.

$\mathbb{C}^{n \times 1} = R(A) \oplus N(A)$ is equivalent to the fact that the index of A is 1 (the result is given in page 154 of [3]), this implies that only matrices of index 1 have pseudoinverse. This problem will be solved by the Drazin inverse, which extends the concept of group inverse to matrices of any index.

If A is a singular square matrix, $\text{Ind}(A) = 1$ is equivalent to the fact that all its 0-vectors have degree 1 (page 161 of [3]), what gives a characterization of the existence of $A^\#$ in terms of spectral properties. We finally give a great result about the spectral properties of $A^\#$ (if it exists):

Theorem 2.

Let A be a square matrix of index 1. Then, $A^\#$ is the only S -inverse of A that satisfies (1) or (2). If A is diagonalizable, $A^\#$ is the only S -inverse of A (proof in page 163 of [3]).

We will not deepen in more properties about the group inverse, the ones that we have studied are enough for a basic understanding of the Drazin inverse, which is the

aim of this section.

Drazin inverse

Drazin inverse is a generalization of the group inverse that tries to avoid its biggest drawback: despite of its great spectral properties, the group inverse only exists for matrices of index 1.

The Drazin inverse is defined with the aim of extending the spectral properties of the group inverse (although they weaken a bit) to matrices of any index.

Theorem 3.

Given a square matrix A of index k , it exists an only pseudoinverse of A denoted by A^D that satisfies (1^k), (2) and (5) (if we take $k = 1$ we obtain the group inverse).

$$\begin{aligned} A^k A^D A &= A^k, \quad (1) & A^D A A^D &= A^D, \quad (2) \\ A A^D &= A^D A, \quad (5) \end{aligned}$$

So generalizing the first equation (in comparison with the group inverse) we achieve existence of this kind of pseudoinverse for any square matrix independently of its index. However, the Theorem 2 tells us that if A^D is a S -inverse, then A and A^D have index 1. This brings us back to the case of the group inverse. So, in order to achieve more generality with this new kind of pseudoinverse, we will have to relax the definition of S -inverse (and consequently the spectral properties of the pseudoinverse).

Definition 4

Given a square matrix A , the pseudoinverse X of A is a S' -inverse if, for any $\lambda \neq 0$ and x complex vector:

x is a λ -vector of A of degree p if and only if x is a $\lambda^\dagger$ -vector of X degree p .

In the case that $\lambda = 0$, x is a 0-vector of A if and only if x is a 0-vector of X (independently of the degree).

With this new definition, for a square matrix A of any index, we can insure the following spectral result:

Theorem 4.

Given any square matrix A , A and A^D are S' -inverses.

In conclusion, we have shown that the Drazin inverse is a pseudoinverse that can be defined in terms of very intuitive spectral properties, whose existence and uniqueness is guaranteed for any square matrix and that matches with the traditional inverse when the matrix is invertible.

Numerical methods to compute Drazin inverse

Methods for matrices with real spectrum

Basic results

Given A a complex matrix, we denote $\sigma(A)$ its spectrum and $\rho(A) = \max\{|\lambda| : \lambda \in \sigma(A)\}$ its spectral radius. We will work with the spectral norm, represented by $\|\cdot\|$ (remember that $\|A\| = \sqrt{\rho(A^*A)}$).

This part of the paper is based in [2]. A basic result given in that article in order to build numerical methods is:

Theorem 5.

Let $A \in \mathbb{C}^{n \times n}$ with $\text{Ind}(A) = k$, then $A^D = \tilde{A}^{-1}A^l$, with $l \geq k$, where $\tilde{A} = A^{l+1}|_{R(A^k)}$.

The restriction $\tilde{A} = A^{l+1}|_{R(A^k)}$ removes the null eigenvalues of the matrix, that is why $\tilde{A}$ is invertible.

The following result enables us to compute the Drazin Inverse using numerical methods.

Theorem 6.

Let $A \in \mathbb{C}^{n \times n}$ with $\text{Ind}(A) = k$ and $\tilde{A} = A^{l+1}|_{R(A^k)}$ for $l \geq k$. Suppose the spectrum of A is real. If Ω is an open set such that $\sigma(\tilde{A}) \subset \Omega \subset (0, +\infty)$ (the eigenvalues of $\tilde{A}$ must be positive) and $S_n : \Omega \rightarrow \mathbb{R}$ is a family of continuous real functions such that $\lim_{n \rightarrow \infty} S_n(x) = \frac{1}{x}$ uniformly in $\sigma(\tilde{A})$. Then:

$$A^D = \lim_{n \rightarrow \infty} S_n(\tilde{A})A^l$$

$$\frac{\|S_n(\tilde{A})A^l - A^D\|_P}{\|A^D\|_P} \leq \max_{x \in \sigma(\tilde{A})} |S_n(x)x - 1| + O(\varepsilon), \quad \varepsilon > 0.$$

The norm $\|\cdot\|_P$ is defined using the spectral norm: $\|A\|_P = \|P^{-1}AP\|$, where P is an invertible matrix such that $P^{-1}AP$ is the Jordan canonical form of A . A basic result used in the proof is that for any $A \in \mathbb{C}^{n \times n}$ and for any $\epsilon > 0$ exists a multiplicative matrix norm $\|\cdot\|_P$ (associated to a matrix P) such that $\|A\|_P \leq \rho(A) + \epsilon$ (can be found at page 21 of [3]).

The previous theorem is the basic result of the first part of the section. We will work following this philosophy: using different numerical methods we will compute approximations of the solution to the equation $y^{-1} - x = 0$ ($\frac{1}{x}$ is the solution). If these

approximation functions $S_n(x)$ converge uniformly (to $\frac{1}{x}$) we can apply the previous result and obtain an approximation of the Drazin inverse. The following result will be very useful in order to give the adequate values to some parameters that appear in some $S_n(x)$:

Proposition 2

Let $A \in \mathbb{C}^{n \times n}$ with $\text{Ind}(A) = k$ and $\tilde{A} = A^{l+1}|_{R(A^k)}$ for $l \geq k$. Then, for each $\lambda \in \sigma(\tilde{A})$:

$$\frac{1}{\|A^D\|^{l+1}} \leq \lambda \leq \|A\|^{l+1}$$

Euler-Knopp method

In this method we work with the family of functions $S_n(x) = \alpha \sum_{j=0}^n (1 - \alpha x)^j$. The sequence converges when $|1 - \alpha x| < 1$, $E_\alpha = \{x \in \mathbb{R} : |1 - \alpha x| < 1\} = \{x \in \mathbb{R} : 0 < \alpha < 2/\alpha\}$ and given that it is a geometric sum:

$$\alpha \sum_{j=0}^{+\infty} (1 - \alpha x)^j = \frac{\alpha}{1 - (1 - \alpha x)} = \frac{1}{x}$$

In E_α the convergence is uniform.

In order to use the Theorem 6, which allows us to compute the Drazin inverse, we need the convergence to be uniform in $\sigma(\tilde{A})$, this satisfies if $\sigma(\tilde{A}) \subset E_\alpha$. In proposition 2 we have seen that $\sigma(\tilde{A}) \subset [1/\|A^D\|^{l+1}, \|A\|^{l+1}] \subset (0, \|A\|^{l+1}]$, to assure that $\sigma(\tilde{A}) \subset E_\alpha$ we take $(0, \|A\|^{l+1}] \subset E_\alpha$, so the parameter must satisfy $0 < \alpha < 2\|A\|^{-(l+1)}$.

Now we are in conditions to apply Theorem 6, the following sequence converges to A^D :

$$S_n(\tilde{A})A^l = \alpha \sum_{j=0}^n (1 - \alpha A^{l+1})^j A^l$$

We write A^{l+1} but remember that this matrix is restricted to $R(A^k)$ by definition of $\tilde{A}$.

Recursive formulas are easier to compute, so we give a recursive formula of this method (immediatly obtained from the formula above):

$$A_0 = \alpha A^l, \quad A_{n+1} = (I - \alpha A^{l+1})A_n + \alpha A^l.$$

Lets study the error bound using Theorem 6 again. We compute $\max_{x \in \sigma(\tilde{A})} |S_n(x)x - 1|$, but first some calculation:

$$S_n(x) = \alpha \frac{1-(1-\alpha x)^{n+1}}{1-(1-\alpha x)} = \frac{1-(1-\alpha)^{n+1}}{x}, \quad S_{n+1}(x) = \frac{1-(1-\alpha)^{n+2}}{x}$$

So $S_{n+1}(x)x - 1 = (1 - \alpha x)S_n(x)$, what recursively implies that: $|S_n(x)x - 1| = (1 - \alpha x)^{n+1}$

So, back to the original problem:

$$\max_{x \in \sigma(\tilde{A})} |S_n(x)x - 1| = \max_{x \in \sigma(\tilde{A})} |1 - \alpha x|^{n+1}$$

If $x \in \sigma(\tilde{A})$ then proposition 2 tells us that $1/\|A^D\|^{l+1} \leq x \leq \|A\|^{l+1}$ (and we already know that $0 < \alpha < 2\|A\|^{-(l+1)}$), so we have that:

$$|1 - \alpha x| \leq \beta := \max\{|1 - \alpha\|A\|^{l+1}|, |1 - \alpha\|A^D\|^{-(l+1)}|\} < 1$$

We can also affirm that $\max_{x \in \sigma(\tilde{A})} |1 - \alpha x|^{n+1} \leq \beta^{n+1} \rightarrow 0$, what assures the convergence of the method. A bound for the error is given by:

$$\frac{\|A_n - A^D\|_P}{\|A^D\|_P} \leq \beta^{n+1} + O(\varepsilon), \quad \varepsilon > 0.$$

Newton-Ralphson method

This method is based in an idea we have already studied: solving the equation $s(y) := y^{-1} - x = 0$ by numerical methods.

The method we will use is the most famous one: Newton-Ralphson method, which uses tangent lines to approximate the roots of $s(y)$. It is defined as follows:

$$y_{n+1} = y_n - s(y_n)/s'(y_n) = y_n(2 - xy_n)$$

One of the inconvenients of this method is that we need a suitable seed y_0 , otherwise we will not have convergence to the root. We will call y_n as $S_n(x)$, so we write $S_{n+1}(x) = S_n(x)(2 - xS_n(x))$. Suppose $\alpha > 0$, we define $S_0(x) = \alpha$ and we will look for suitable conditions to obtain convergence. Note that:

$$xS_{n+1}(x) - 1 = x(S_n(x)(2 - xS_n(x))) - 1 = 2xS_n(x) - x^2(S_n(x))^2 - 1 = -(1 - xS_n(x))^2$$

By iteration we get:

$$|xS_n(x) - 1| = |xS_0(x) - 1|^{2^n} = |x\alpha - 1|^{2^n}$$

$$\max_{x \in \sigma(\tilde{A})} |S_n(x)x - 1| = \max_{x \in \sigma(\tilde{A})} |1 - \alpha x|^{2^n}$$

Applying the same reasoning as above (we take $0 < \alpha < 2\|A\|^{-(l+1)}$), we obtain that:

$$|1 - \alpha x| \leq \beta := \max\{|1 - \alpha\|A\|^{l+1}|, |1 - \alpha\|A^D\|^{-(l+1)}|\} < 1$$

We have that $\max_{x \in \sigma(\tilde{A})} |1 - \alpha x|^{2^n} \leq \beta^{2^n} \rightarrow 0$, what assures the convergence of the method (it is quadratic, better than the previous method). As it is said in the Theorem 6, $A^D = \lim_{n \rightarrow +\infty} S_n(\tilde{A})A^l$, we define $A_n = S_n(\tilde{A})A^l$ and given: $S_0(\tilde{A}) = \alpha I$, $S_{n+1}(\tilde{A}) = S_n(\tilde{A})(2I - \tilde{A}S_n(\tilde{A}))$, we obtain the following method (assuming the restriction to $R(A^k)$ of $\tilde{A} = A^{l+1}$):

$$A_0 = \alpha A^l, \quad A_{n+1} = A_n(2I - AA_n).$$

As we have seen, the error bound for this method is:

$$\frac{\|A_n - A^D\|_P}{\|A^D\|_P} \leq \beta^{2^n} + O(\varepsilon), \quad \varepsilon > 0.$$

Methods for matrices with complex spectrum

Basic results

The aim of this section is to extend the results obtained in the previous one to matrices with complex spectrum. In order to do that, we modify Theorem 6:

Theorem 7.

Let $A \in \mathbb{C}^{n \times n}$ with $\text{Ind}(A) = k$, let $\tilde{A} = A^{l+1}|_{R(A^k)}$ for $l \geq k$. If G is an open set excluding origin such that $\sigma(\tilde{A}) \subset G$ and $S_n : G \rightarrow \mathbb{R}$ a polynomial sequence such that $S_m(z) \rightarrow \frac{1}{z}$ on any compact subset of G (in compacts convergence is uniform), then:

$$A^D = \lim_{m \rightarrow +\infty} S_m A^l$$

Now, following an analogous structure to the previous section, we have to give a similar result to Proposition 2 for the complex case:

Proposition 3

Let $A \in \mathbb{C}^{n \times n}$ with $\text{Ind}(A) = k$ and $\tilde{A} = A^{l+1}|_{R(A^k)}$ for $l \geq k$. If $\sigma(\tilde{A})$ is in the right open half-plane and if

$$\phi = \max_{\lambda \in \sigma(\tilde{A})} \{|\arg \lambda| : -\pi/2 < \phi < \pi/2\} \text{ and } 0 < \alpha < \frac{2 \cos \phi}{\|A\|^{l+1}},$$

then

$$\sigma(\tilde{A}) \subset \{z \in \mathbb{C} : |1 - \alpha z| < 1\} =: E_\alpha.$$

$$\beta_1 := \max_{z \in \sigma(\tilde{A})} |1 - \alpha z| \leq \max_{j=1,2} \{1 - \alpha r_j e^{i\phi}\} < 1,$$

where $r_1 = 1/\|A^D\|^{l+1}$ and $r_2 = \|A\|^{l+1}$.

If $\sigma(\tilde{A}) \subset (0, +\infty)$ we are back to the case studied in the previous section, where $|1 - \alpha x| \leq \max\{|1 - \alpha r_1|, |1 - \alpha r_2|\} < 1$.

Given this results, [1] extends Euler-Knobb and Newton-Ralphson methods to the case $\sigma(\tilde{A})$ is in the right open half-plane.

Euler-Knobb

We have already studied the following method:

$$A_0 = \alpha A^l, \quad A_{m+1} = (I - \alpha A^{l+1})A_m + \alpha A^l.$$

If we assume that we are under the hypothesis of proposition 3, we can reproduce the same arguments given for the real case:

$$\max_{z \in \sigma(\tilde{A})} |S_m(z)z - 1| = \max_{z \in \sigma(\tilde{A})} |1 - \alpha z|^{n+1} = \beta_1^{n+1} \rightarrow 0$$

so using Theorem 7 we conclude that A_m converges to A^D . In the proof of Theorem 6 we can find an argument that allows us to extend an error bound for this case. Applying it, we have that for any $\epsilon \in (0, 1 - \beta_1)$, there is a norm $\|\cdot\|_P$ such that $\|A_m - A^D\|_P < \|A^D\|_P \beta^m$, where $\beta = \epsilon + \beta_1 < 1$.

Newton-Ralphson

Analagously, if we assume that we are under the hypothesis of proposition 3, reproducing the real case:

$$\max_{z \in \sigma(\tilde{A})} |S_n(z)z - 1| = \max_{z \in \sigma(\tilde{A})} |1 - \alpha z|^{2^n} = \beta_1^{2^n} \rightarrow 0.$$

so using Theorem 7 we conclude that A_m converges to A^D . Following the same reasoning as above, for any $\epsilon \in (0, 1 - \beta_1)$, there is a norm $\|\cdot\|_P$ such that $\|A_m - A^D\|_P < \|A^D\|_P \beta^{2^m}$, where $\beta = \epsilon + \beta_1 < 1$.

Numerical experiments

In this section we will show some numerical experiments in order to put into practice the methods that we have developed. This will strengthen our comprehension of the methods.

Real case

First, we will study the real case. We need an error bound in order to set a tolerance that helps us to finish the algorithm. Remember that:

$$\frac{\|S_n(\tilde{A})A^l - A^D\|_P}{\|A^D\|_P} \leq \max_{x \in \sigma(\tilde{A})} |S_n(x)x - 1| + O(\epsilon), \quad \epsilon > 0.$$

We also defined a β that, when $0 < \alpha < 2\|A\|^{-(l+1)}$, helped us to prove the convergence (since the expression $|1 - \alpha x|$ was involved in the error bound):

$$|1 - \alpha x| \leq \beta := \max\{|1 - \alpha\|A\|^{l+1}|, |1 - \alpha\|A^D\|^{-(l+1)}|\} < 1$$

However, computing $\|A^D\|$ without knowing A^D is not a simple task, so in practice we will work with $\beta := \max_{x \in \sigma(\tilde{A})} |1 - \alpha x|$ (it is a lower bound than the previous one).

Remember that in Euler-Knobb:

$$\max_{x \in \sigma(\tilde{A})} |S_n(x)x - 1| = \max_{x \in \sigma(\tilde{A})} |1 - \alpha x|^{n+1}$$

$$A_0 = \alpha A^l, \quad A_{n+1} = (I - \alpha A^{l+1})A_n + \alpha A^l.$$

In Newton-Raphson:

$$\max_{x \in \sigma(\tilde{A})} |S_n(x)x - 1| = \max_{x \in \sigma(\tilde{A})} |1 - \alpha x|^{2^n}$$

$$A_0 = \alpha A^l, \quad A_{n+1} = A_n(2I - AA_n).$$

We will follow this steps to apply the methods to a matrix A :

1. Compute the index k of A (we need $l \geq k$) and $\|A\|$ in order to obtain a bound for α .
2. As shown in the proof of Theorem 6 (if $\lambda \in \sigma(A)$, then $\lambda^n \in \sigma(A^n)$), we choose $l \in \mathbb{N}$ so $\sigma(A^{l+1}) \subset [0, +\infty)$ and consequently $\sigma(\tilde{A}) \subset (0, +\infty)$.
3. Given a value of tolerance tol , we compute $n, m \in \mathbb{N}$ the minimum number of iterations in order to satisfy $\beta^{n+1} < tol$ (Euler) and $\beta^{2^m} < tol$ (Newton).
4. Now we are in conditions to compute the solutions given by the methods (the code of the functions used are in the appendix).

Complex case

This case is analogous to the real one, but with some particularities. We will use as error bound the same values: β^{n+1} (Euler) and β^{2^n} (Newton) with $\beta := \max_{z \in \sigma(\tilde{A})} |1 - \alpha z|$. Now we take $0 < \alpha < 2 \cos \phi \|A\|^{-(l+1)}$ with $\phi = \max_{\lambda \in \sigma(\tilde{A})} \{|\arg \lambda| : -\pi/2 < \phi < \pi/2\}$.

We want $\sigma(\tilde{A})$ to be contained in the right open half-plane (we know that given $z = r(\cos(\theta) + i \sin(n\theta))$, then $z^n = r^n(\cos(n\theta) + i \sin(n\theta))$). Using this expression we can find a $l \geq k$ that satisfy the wanted condition (at least, when all the arguments are rationally proportional to π).

Given $\sigma(\tilde{A}) = \{z_1, \dots, z_k\}$ such that $\theta_i = \frac{p_i}{q_i} \pi$ (with an irreducible fraction). Let $m = \text{lcm}(\theta_1, \dots, \theta_k)$ and $n = 2m$, then $\theta_i n = 2 \frac{p_i}{q_i} m \pi = 2s\pi$ with $s \in \mathbb{N}$, so $\cos(\theta_i n) = 1$.

In the case that some $\theta_j \neq \frac{p_j}{q_j} \pi$, we can try a proof based in the density argument $\{\theta_i n \bmod 2\pi : n \in \mathbb{N}\} = [0, 2\pi)$, but it is not clear that this path always works.

Examples

Example 1

First we will study a matrix with only real eigenvalues.

$$A = \begin{bmatrix} 3 & -1 & -1 & -1 & 1 & -3 & 1 & 2 \\ 3 & -1 & -1 & -1 & 1 & -3 & 1 & 2 \\ 2 & 0 & -1 & -1 & 1 & -3 & 1 & 2 \\ 1 & 0 & 0 & -1 & 1 & -3 & 1 & 2 \\ 0 & 0 & 0 & 0 & 1 & -3 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 & -2 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

We have that $\sigma(A) = \{0, 0, 0, 0, 1, 1, -1, -2\}$ (we have negative eigenvalues). The index of the matrix is $k = 4$, so choosing $l = 5$ is enough to obtain only positive eigenvalues for $\tilde{A} = A^{l+1}|_{R(A^k)}$. If we compute the upper bound for α we obtain $\alpha < \|A\|^{l+1} = \|A\|^6 = 1.67 \cdot 10^{-6}$. However, since we are not choosing the same β as in the theoretical construction, it is enough to choose an α that makes $\beta := \max_{x \in \sigma(\tilde{A})} |1 - \alpha x| < 1$. Let $\alpha = 0.001$ (a bigger α makes the method converge fastly). If we set the tolerance at $tol = 10^{-12}$, we need 27617 iterations with Euler-Knopp and 15 iterations with Newton-Raphson to obtain an approximation of the Drazin inverse:

$$A^D = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & -3/2 & -1/2 & 2 \\ 0 & 0 & 0 & 0 & 1 & -3/2 & -1/2 & 2 \\ 0 & 0 & 0 & 0 & 1 & -3/2 & -1/2 & 2 \\ 0 & 0 & 0 & 0 & 1 & -3/2 & -1/2 & 2 \\ 0 & 0 & 0 & 0 & 1 & -3/2 & -1/2 & 2 \\ 0 & 0 & 0 & 0 & 0 & -1/2 & -1/2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Example 2

Now we study a matrix with complex eigenvalues.

$$B = \begin{bmatrix} 2 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 & -1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 2 \end{bmatrix}$$

We have that $\sigma(B) = \{0, 0, 3/2 + \sqrt{3}/6i, 3/2 - \sqrt{3}/6i, 3/2 + \sqrt{3}/6i, 3/2 - \sqrt{3}/6i, 2, 2\}$. The index of the matrix is $k = 2$ and if we choose $l = 2$ too, all the eigenvalues are located in the open right half-plane. The theoretical bound for α we obtain 0.0533, so choosing $\alpha = 0.01$ is enough. If we set the same tolerance $tol = 10^{-12}$, Euler-Knopp needs 913 iterations to converge while Newton Raphson only needs 10. The approximation of the Drazin inverse obtained with Newton is:

$$\begin{bmatrix} 0.4286 & -0.1429 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.4286 & 0.8571 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.6735 & -0.4898 & 0.25 & -0.25 & 0 & 0 & 0 & 0 \\ -0.6735 & -0.4898 & -0.25 & 0.25 & 0 & 0 & 0 & 0 \\ 0.9633 & 0.7022 & 0.3735 & -0.0499 & 0.25 & -0.25 & -0.4898 & -0.6735 \\ 0.9633 & 0.7022 & 0.3735 & -0.0499 & -0.25 & 0.25 & -0.4898 & -0.6735 \\ -1.1545 & -0.9009 & -0.5510 & -0.1224 & 0 & 0 & 0.8571 & 0.4286 \\ 0.2536 & 0.2012 & 0.1276 & 0.0561 & 0 & 0 & -0.1429 & 0.4286 \end{bmatrix}$$

Because of the disparity between the number of iterations for each method, graphics that compare the rate of convergence of the methods are not very visual. Newton-Raphson is a very fast method that converges quadratically (proved theoretically) while the performance of Euler-Knobb is not very good in general (we have seen that converges like β^n).

Conclusions

In this work, we have studied the spectral structure of the Drazin inverse, which has enabled us to develop two iterative methods for its computation. We first considered the case of real matrices and subsequently extended the results to complex matrices. The performance of these methods has proved to be quite satisfactory. Nevertheless, a possible direction for future research is to improve their efficiency, particularly when dealing with large-scale matrices.

Acknowledgements

The authors would like to thank the Universitat de València and the Universitat Politècnica de València for organizing the XIII Congress of the Master in Mathematical Research and Doctorate in Mathematics.

References

- [1] X. Li and Y. Wei. Iterative methods for the Drazin inverse of a matrix with a complex spectrum. *Applied Mathematics and Computation*, 147(3):855–862, 2004. Elsevier.
- [2] Y. Wei and H. Wu. The representation and approximation for Drazin inverse. *Journal of Computational and Applied Mathematics*, 126(1–2):417–432, 2000. Elsevier.
- [3] A. Ben-Israel and T. N. E. Greville. Generalized Inverses: Theory and Applications. *CMS Books in Mathematics*, Springer New York, 2006.

Morphisms of rewriting systems

Juan Climent Vidal¹, Enric Cosme Llópez^{2,3}, and Raúl Ruiz Mora^{*2}

¹*Departament de Lògica i Filosofia de la Ciència, Universitat de València, Av. Vicent Andrés Estellés 19, 46100–Burjassot, Spain*

²*Departament de Matemàtiques, Universitat de València, Av. Vicent Andrés Estellés 19, 46100–Burjassot, Spain*

³*Nantong University, School of Mathematics and Statistics, China*

Abstract

We develop a tiered definition for morphisms between rewriting systems. At layer 0, which encompasses the underlying algebraic basis—operations, and variables—we employ the concept of a derivor to formalize the notion of a zeroth-order morphism. We introduce the extension of this definition to terms, subsequently constructing the category of zeroth-order rewriting systems. This framework is extended to layer 1 by incorporating rewriting rules; here, a morphism is defined via the preceding zeroth-order morphism coupled with a mapping that assigns the rewriting rules in the domain to paths within the codomain. We show that this mapping can be extended to paths and that this extension is compatible with path classes, leading to the construction of the category of rewriting systems. Finally, this inductive process is iterated at layer 2 to account for second-order rewriting rules. A second-order morphism is characterized by a morphism of rewriting systems and a mapping that assigns second-order rewriting rules in the domain to second-order paths in the codomain. By extending this to second-order paths and ensuring compatibility with second-order path classes, we construct the category of second-order rewriting systems.

Introduction

In the first part of this work [9] we introduced $\text{Pth}_{\mathcal{A}}$ the set of paths associated with a rewriting system $\mathcal{A} = (\Sigma, X, \mathcal{A})$ and equipped it with a structure of partial $\Sigma^{\mathcal{A}}$ -algebra, a structure of category, and a structure of Artinian ordered set. Moreover, we identified $[\text{PT}_{\mathcal{A}}]$, a subquotient of $T_{\Sigma^{\mathcal{A}}}(X)$, that is isomorphic to the algebraic, categorical, and ordered structures on $[\text{Pth}_{\mathcal{A}}]$. This constituted a Curry-Howard type isomorphism. Additionally, we proved that these two structures are isomorphic to $T_{\mathcal{E}^{\mathcal{A}}}(\text{Pth}_{\mathcal{A}})$, the free partial $\Sigma^{\mathcal{A}}$ -algebra in the variety of partial algebras $\text{PAlg}(\mathcal{E}^{\mathcal{A}})$. Similarly, in the second part of this work [9] we introduced $\text{Pth}_{\mathcal{A}^{(2)}}$ the set of second-

*raul.ruiz-mora@uv.es

order paths associated with a second-order rewriting system $\mathcal{A}^{(2)} = (\mathcal{A}, \mathcal{A}^{(2)})$ and equipped it with a structure of partial $\Sigma^{\mathcal{A}^{(2)}}$ -algebra, a structure of category, and a structure of Artinian ordered set. Moreover, we identified $\llbracket \text{PT}_{\mathcal{A}^{(2)}} \rrbracket$, a subquotient of $T_{\Sigma^{\mathcal{A}^{(2)}}}(X)$, that is isomorphic to the algebraic, categorical, and ordered structures on $\llbracket \text{Pth}_{\mathcal{A}^{(2)}} \rrbracket$. This constituted a Curry-Howard type isomorphism. Additionally, we proved that these two structures are isomorphic to $T_{\mathcal{E}^{\mathcal{A}^{(2)}}}(\text{Pth}_{\mathcal{A}^{(2)}})$, the free partial $\Sigma^{\mathcal{A}^{(2)}}$ -algebra in the variety of partial algebras $\text{PAI}(\mathcal{E}^{\mathcal{A}^{(2)}})$.

In this part, we start by defining $\text{Rws}_{\delta}^{(0)}$, the category of zeroth order rewriting systems, which is at the basis of all the work. Let Sig_{δ} be the category whose objects are the signatures, i.e., families $\Sigma = (\Sigma_n)_{n \in \mathbb{N}}$ of operation symbols, and whose morphisms from the signature Σ to the signature Λ are the derivors $c = (c_n)_{n \in \mathbb{N}}$ (see [8]). Let Alg_{δ} be the category whose objects are the algebras, i.e., the ordered pairs $(\Sigma, \mathbf{A})$, where Σ is a signature and $\mathbf{A}$ a Σ -algebra, and whose morphisms from the algebra $(\Sigma, \mathbf{A})$ to the algebra $(\Lambda, \mathbf{B})$ are the ordered pairs (c, f) , with c a derivor from Σ to Λ and f a homomorphism of Σ -algebras from $\mathbf{A}$ to $c_{\delta}^*(\mathbf{B})$, where c_{δ}^* is the functor from $\text{Alg}(\Lambda)$ to $\text{Alg}(\Sigma)$ determined by the derivor c (see [8]).

Then $\text{Rws}_{\delta}^{(0)}$ is the category whose objects are the ordered pairs (Σ, X) , where Σ is a signature and X a set, which we will call zeroth order rewriting systems, and whose morphisms from the zeroth order rewriting system $\mathcal{A}^{(0)} = (\Sigma, X)$ to the zeroth order rewriting system $\mathcal{B}^{(0)} = (\Lambda, Y)$ are the ordered triples $((\Sigma, X), f^{(0)}, (\Lambda, Y))$, denoted by $f^{(0)}: \mathcal{A}^{(0)} \rightarrow \mathcal{B}^{(0)}$ for short, in which $f^{(0)} = (c, f^{(0)})$, where

1. c is a derivor from Σ to Λ and
2. $f^{(0)}$ is a mapping from X to $T_{\Lambda}(Y)$.

Note that $T_{\Lambda}(Y)$ is the underlying set of the Σ -algebra $c_{\delta}^*(T_{\Lambda}(Y))$. Let us also note that there exists a full embedding of $\text{Rws}_{\delta}^{(0)}$ into Alg_{δ} , because to give a mapping $f^{(0)}$ from X to $T_{\Lambda}(Y)$ is, naturally, equivalent to give a Σ -homomorphism $f^{(0)\sharp}$ from $T_{\Sigma}(X)$ to $c_{\delta}^*(T_{\Lambda}(Y))$, obtained by the universal property of $T_{\Sigma}(X)$. Actually, we prove that $\text{Rws}_{\delta}^{(0)}$ is isomorphic to $\text{Tw}_{\delta}^{(0)}$, the full subcategory of Alg_{δ} determined by the algebras of the form $(\mathcal{A}^{(0)}, T_{\Sigma}(X))$, with $\mathcal{A}^{(0)}$ varying over the zeroth-order rewriting systems, what we have called in this work zeroth-order towers.

We next consider rewriting systems and morphisms between them. Let $\mathcal{A}^{(1)} = (\Sigma, X, \mathcal{A}^{(1)}) = (\mathcal{A}^{(0)}, \mathcal{A}^{(1)})$ and $\mathcal{B}^{(1)} = (\Lambda, Y, \mathcal{B}^{(1)}) = (\mathcal{B}^{(0)}, \mathcal{B}^{(1)})$ be rewriting systems. A morphism from $\mathcal{A}^{(1)}$ to $\mathcal{B}^{(1)}$ will be an ordered triple $(\mathcal{A}^{(1)}, f^{(1)}, \mathcal{B}^{(1)})$, denoted by $f^{(1)}: \mathcal{A}^{(1)} \rightarrow \mathcal{B}^{(1)}$ or $f^{(1)}$ for short, in which $f^{(1)}$ is an ordered pair $(f^{(0)}, f^{(1)})$ where

1. $f^{(0)} = (c, f^{(0)})$ is a morphism from $\mathcal{A}^{(0)}$ to $\mathcal{B}^{(0)}$ and
2. $f^{(1)}$ is a mapping from $\mathcal{A}^{(1)}$ to $\text{Pth}_{\mathcal{B}^{(1)}}$ such that, for every $p = (M, N) \in \mathcal{A}^{(1)}$, we have that $f^{(1)}(p) \in \text{Pth}_{\mathcal{B}^{(1)}}(f^{(0)\sharp}(M), f^{(0)\sharp}(N))$.

From this definition we obtain, by recursion on $(\text{Pth}_{\mathcal{A}^{(1)}}, \leq_{\text{Pth}_{\mathcal{A}^{(1)}}})$, a Σ -homomorphism $f^{(1)\flat}$ from $\text{Pth}_{\mathcal{A}^{(1)}}^{(0,1)}$ to $c_{\delta}^*(\text{Pth}_{\mathcal{B}^{(1)}}^{(0,1)})$ such that $f^{(1)\flat}$ coincides with $f^{(0)\sharp}$ when restricted

to the $(1, 0)$ -identity paths and with $f^{(1)}$ when restricted to the echelons, i.e., to the one-step paths canonically associated to the rewrite rules on $\mathcal{A}^{(1)}$. We next provide $[\text{Pth}_{\mathcal{B}^{(1)}}]$, i.e., the underlying set of the Σ -algebra $\mathbf{c}_\partial^*([\text{Pth}_{\mathcal{B}^{(1)}}^{(0,1)}])$, with a structure of partial $\Sigma^{\mathcal{A}^{(1)}}$ -algebra, that we denote by $[\text{Pth}_{\mathcal{B}^{(1)}}^{f^{(1)}}]$, and we prove that it belongs to the QE-variety determined by $\mathcal{E}^{\mathcal{A}^{(1)}}$. Since the partial $\Lambda^{\mathcal{B}^{(1)}}$ -algebras $[\text{PT}_{\mathcal{B}^{(1)}}]$ and $\mathbf{T}_{\mathcal{E}^{\mathcal{B}^{(1)}}}(\text{Pth}_{\mathcal{B}^{(1)}})$ are isomorphic to $[\text{Pth}_{\mathcal{B}^{(1)}}]$, an analogous statement holds for the respective partial $\Sigma^{\mathcal{A}^{(1)}}$ -algebras $[\text{PT}_{\mathcal{B}^{(1)}}^{f^{(1)}}]$ and $\mathbf{T}_{\mathcal{E}^{\mathcal{B}^{(1)}}}^{f^{(1)}}(\text{Pth}_{\mathcal{B}^{(1)}})$, built following a similar approach. Thus, since we have proved that $[\text{Pth}_{\mathcal{A}^{(1)}}]$ is isomorphic to $\mathbf{T}_{\mathcal{E}^{\mathcal{A}^{(1)}}}(\text{Pth}_{\mathcal{A}^{(1)}})$ (or to $[\text{PT}_{\mathcal{A}^{(1)}}]$), we obtain, by the universal property of it, a unique $\Sigma^{\mathcal{A}^{(1)}}$ -homomorphism from $[\text{Pth}_{\mathcal{A}^{(1)}}]$ to $[\text{Pth}_{\mathcal{B}^{(1)}}^{f^{(1)}}]$ (from $\mathbf{T}_{\mathcal{E}^{\mathcal{A}^{(1)}}}(\text{Pth}_{\mathcal{A}^{(1)}})$ to $\mathbf{T}_{\mathcal{E}^{\mathcal{B}^{(1)}}}^{f^{(1)}}(\text{Pth}_{\mathcal{B}^{(1)}})$ or from $[\text{PT}_{\mathcal{A}^{(1)}}]$ to $[\text{PT}_{\mathcal{B}^{(1)}}^{f^{(1)}}]$), which we denote by $f^{[1]@}$, satisfying

$$f^{[1]@} \circ \text{pr}_{\mathcal{A}^{(1)}}^{\text{Ker}(\text{CH}^{(1)})} = \text{pr}_{\mathcal{B}^{(1)}, \varphi}^{\text{Ker}(\text{CH}^{(1)})} \circ f^{(1)\flat}.$$

Now, with respect to rewriting systems and morphisms on it, we prove that it does not form a category since, among other things, the natural composition of morphisms is not associative. To overcome this problem, we introduce an equivalence relation on the morphisms. Given two morphisms $\mathbf{f}^{(1)} = (\mathbf{f}^{(0)}, f^{(1)})$ and $\mathbf{g}^{(1)} = (\mathbf{g}^{(0)}, g^{(1)})$, from $\mathcal{A}^{(1)}$ to $\mathcal{B}^{(1)}$, we say that $\mathbf{f}^{(1)}$ is equivalent to $\mathbf{g}^{(1)}$, written $\mathbf{f}^{(1)} \cong^{(1)} \mathbf{g}^{(1)}$, if and only if, $\mathbf{f}^{(0)} = \mathbf{g}^{(0)}$ and $\text{pr}_{\mathcal{B}^{(1)}}^{\text{Ker}(\text{CH}^{(1)})} \circ f^{(1)\flat} = \text{pr}_{\mathcal{B}^{(1)}}^{\text{Ker}(\text{CH}^{(1)})} \circ g^{(1)\flat}$. We prove that, for equivalent morphisms $\mathbf{f}^{(1)} \cong^{(1)} \mathbf{g}^{(1)}$, it holds that $f^{[1]@} = g^{[1]@}$ and the respective partial $\Sigma^{\mathcal{A}^{(1)}}$ -algebras $[\text{Pth}_{\mathcal{B}^{(1)}}^{f^{(1)}}]$ and $[\text{Pth}_{\mathcal{B}^{(1)}}^{g^{(1)}}]$ are equal. This allows us to construct the category $\text{Rws}_\partial^{[1]}$, whose objects are rewriting systems and morphisms are $\cong^{(1)}$ -classes of morphisms of rewriting systems. Actually, we prove that $\text{Rws}_\partial^{[1]}$ is a category and is isomorphic to the category of towers, denoted by $\text{Tw}_\partial^{[1]}$, determined by the partial algebras of the form $(\mathcal{A}^{(1)}, \mathbf{T}_{\mathcal{E}^{\mathcal{A}^{(1)}}}(\text{Pth}_{\mathcal{A}^{(1)}}))$, with $\mathcal{A}^{(1)}$ varying over the rewriting systems.

Finally, we next consider second-order rewriting systems and second-order morphisms between them. Let $\mathcal{A}^{(2)} = (\Sigma, X, \mathcal{A}^{(1)}, \mathcal{A}^{(2)}) = (\mathcal{A}^{(1)}, \mathcal{A}^{(2)})$ and $\mathcal{B}^{(2)} = (\Lambda, Y, \mathcal{B}^{(1)}, \mathcal{B}^{(2)}) = (\mathcal{B}^{(1)}, \mathcal{B}^{(2)})$ be second-order rewriting systems. A morphism from $\mathcal{A}^{(2)}$ to $\mathcal{B}^{(2)}$ will be an ordered triple $(\mathcal{A}^{(2)}, \mathbf{f}^{(2)}, \mathcal{B}^{(2)})$, denoted by $\mathbf{f}^{(2)}: \mathcal{A}^{(2)} \rightarrow \mathcal{B}^{(2)}$ or $\mathbf{f}^{(2)}$ for short, in which $\mathbf{f}^{(2)}$ is an ordered pair $(\mathbf{f}^{(1)}, f^{(2)})$ where

1. $\mathbf{f}^{(1)} = (\mathbf{f}^{(0)}, f^{(1)})$ is a morphism from $\mathcal{A}^{(1)}$ to $\mathcal{B}^{(1)}$ and
2. $f^{(2)}$ is a mapping from $\mathcal{A}^{(2)}$ to $\text{Pth}_{\mathcal{B}^{(2)}}$ such that, for every $\mathbf{p}^{(2)} = ([M], [N]) \in \mathcal{A}^{(2)}$, we have that $f^{(2)}(\mathbf{p}^{(2)}) \in \text{Pth}_{\mathcal{B}^{(2)}}(f^{[1]@}([M]), f^{[1]@}([N]))$.

From this definition we obtain, in a similar manner, all the results introduced so far, thus obtaining the category $\text{Rws}_\partial^{[2]}$, whose objects are second-order rewriting systems and morphisms are $\cong^{(2)}$ -classes of morphisms of second-order rewriting systems. We also prove that $\text{Rws}_\partial^{[2]}$ is a category and is isomorphic to the category of second-order towers, denoted by $\text{Tw}_\partial^{[2]}$, determined by the partial algebras of the form $(\mathcal{A}^{(2)}, \mathbf{T}_{\mathcal{E}^{\mathcal{A}^{(2)}}}(\text{Pth}_{\mathcal{A}^{(2)}}))$, with $\mathcal{A}^{(2)}$ varying over the second-order rewriting systems.

In this paper, next to each result, the reader will find the corresponding reference to the complete proof in [9]. Our work is framed in the study of morphisms of rewriting systems in the context of many-sorted algebras. Nevertheless, to facilitate understanding, in this paper we have opted to present the single-sorted version of our findings.

The only prerequisites for reading this work are familiarity with category theory [15, 16], universal algebra [1, 4, 5, 6, 7, 13, 14, 17, 18, 19], the theory of ordered sets [2, 11] and set theory [3, 12]. The reader is advised to consult in [9], the first and second parts of this project.

Preliminaries

In this paper we will use the notion of derivor (see [8]) as ground for our definition of morphism of rewriting systems. We begin this section by introducing the concept of clone.

Definition 1

A clone is a triple $\mathbf{C} = ((C_n)_{n \in \mathbb{N}}, ((\pi_{k,n})_{k \in n})_{n \in \mathbb{N}}, (\circ_{m,n})_{m,n \in \mathbb{N}})$ where

- For every $n \in \mathbb{N}$, C_n is a set.
- For every $k, n \in \mathbb{N}$ with $k \in n$, $\pi_{k,n} \in C_n$.
- For every $m, n \in \mathbb{N}$, $\circ_{m,n}: C_m \times (C_n)^m \longrightarrow C_n$. We agree to use only $\circ$ leaving the index m and n implicit.

satisfying that

1. For every $n \in \mathbb{N}$ and every $c \in C_n$, $c \circ (\pi_{i,n})_{i \in n} = c$;
2. For every $k, n, m \in \mathbb{N}$ with $k \in n$ and every $(c_i)_{i \in n} \in C_m^n$, $\pi_{k,n} \circ (c_i)_{i \in n} = c_k$; and
3. For every $m, n, p \in \mathbb{N}$, every $c \in C_m$, every $(d_i)_{i \in m} \in C_n^m$ and every $(e_j)_{j \in n} \in C_p^n$,
 $c \circ ((d_i \circ (e_j)_{j \in n})_{i \in m}) = (c \circ (d_i)_{i \in m}) \circ (e_j)_{j \in n}$.

A morphism from the clone $\mathbf{C}$ to the clone $\mathbf{C}'$ is a family $f = (f_n)_{n \in \mathbb{N}}$ where, for every $n \in \mathbb{N}$, $f_n: C_n \longrightarrow C'_n$ such that

1. For every $k, n, m \in \mathbb{N}$ with $k \in n$ $f_n(\pi_{k,n}) = \pi'_{k,n}$; and
2. For every $m, n \in \mathbb{N}$, every $c \in C_m$, every $(d_i)_{i \in m} \in C_n^m$, $f_n(c \circ (d_i)_{i \in m}) = f_m(c) \circ (f_n(d_i))_{i \in m}$.

In the next two propositions we introduce a representation of the free clone.

Proposition 1

[9, Proposition 2.10.6] Let Σ be a signature and let $V = \{v_i \mid i \in \mathbb{N}\}$ be a denumerable set of variables. Then

- The family $T_\Sigma(V) = (T_\Sigma(V_n))_{n \in \mathbb{N}}$ where $V_n = \{v_i \mid i \in n\}$,
- the family $((v_k)_{k \in n})_{n \in \mathbb{N}}$, and
- the operation $P \circ (Q_i)_{i \in m} = \left((Q_i)_{i \in m} \right)^\# (P) \in T_\Sigma(V_n)$,

is a clone that we will denote by $T_\Sigma(V)$.

Proposition 2

[9, Proposition 2.10.11] Let $V = \{v_i \mid i \in \mathbb{N}\}$. For every signature Σ , the pair $(\eta^\Sigma, T_\Sigma(V))$, where η^Σ is the canonical inclusion of Σ into the clone $T_\Sigma(V)$ which assigns, to every operation symbol σ in Σ_n , the term $\sigma((v_i)_{i \in n})$, has the following universal property: for every clone C and every family of mappings $f = (f_n)_{n \in \mathbb{N}}: \Sigma \longrightarrow C$, there exists a unique morphism of clones $f^\#: T_\Sigma(V) \longrightarrow C$ such that $f^\# \circ \eta^\Sigma = f$, that is, for every $n \in \mathbb{N}$, $f_n^\# \circ \eta_n^\Sigma = f_n$.

We next introduce the concept of derivor from one signature to another.

Definition 2

Let Σ and Λ be two signatures. A derivor from Σ to Λ is a family $c = (c_n)_{n \in \mathbb{N}}$ where, for every $n \in \mathbb{N}$,

$$c_n: \Sigma_n \longrightarrow T_\Lambda(V_n)$$

and $V_n = \{v_i \mid i \in n\}$. Hence, for every $n \in \mathbb{N}$ and every $\sigma \in \Sigma_n$, $c_n(\sigma) \in T_\Lambda(V_n)$, i.e., $c_n(\sigma)$ is a term with variables in V_n and operation symbols in Λ .

Remark. While derivors are not the most general type of morphism that might be considered between signatures—for instance, one could consider polyderivors, see [10]—, they are an important class of such morphisms. One reason for its relevance is its formal properties (see below), another that there are many mathematical examples of them which are of interest (see [10]).

Moreover, we show that signatures and derivors constitute a category.

Definition 3

Let $c: \Sigma \longrightarrow \Lambda$ and $d: \Lambda \longrightarrow \Omega$ be derivors. Then $d \circ c = (d_n)_{n \in \mathbb{N}} \circ (c_n)_{n \in \mathbb{N}}$, the composition of c and d , is the derivor $(d_n^\# \circ c_n)_{n \in \mathbb{N}}$, where $d_n^\# \circ c_n$ is the mapping from Σ_n to $T_\Omega(V_n)$, being $d_n^\#$ the n -th component of the canonical extension of d obtained in Proposition 2.

For every signature Σ , the identity derivor at Σ is $\eta^\Sigma = (\eta_n^\Sigma)_{n \in \mathbb{N}}$, the canonical inclusion of Σ into the clone $T_\Sigma(X)$.

Proposition 3

[9, Proposition 2.10.18] Signatures and derivors constitute a category denoted by Sig_δ .

Finally, we assign to every derivor a functor which transforms Λ -algebras into Σ -algebras. This assignment is the morphism part of a functor Alg_δ from Sig_δ to Cat . We have not defined this functor explicitly because it is not strictly necessary in the presentation of this work.

Proposition 4

[9, Proposition 2.10.21] Let $c: \Sigma \longrightarrow \Lambda$ be a derivor. Then $\mathbf{c}_\delta^*$ is the functor from $\text{Alg}(\Lambda)$ to $\text{Alg}(\Sigma)$ that sends

1. every Λ -algebra $\mathbf{B} = (B, G)$ to $\mathbf{c}_\delta^*(\mathbf{B}) = (B, G^\sharp \circ d)$ where $G^\sharp$ is the canonical extension of G obtained from Proposition 2; and
2. every Λ -homomorphism f from $\mathbf{B}$ to $\mathbf{B}'$ to the homomorphism f from $\mathbf{c}_\delta^*(\mathbf{B})$ to $\mathbf{c}_\delta^*(\mathbf{B}')$.

Zeroth-order morphisms

In this section we begin by introducing the notions of zeroth-order rewriting systems and that of zeroth-order morphism based on the notion of derivor.

Definition 4

A zeroth-order rewriting system is an ordered pair $\mathcal{A}^{(0)} = (\Sigma, X)$ where Σ is a signature and X is a set. Let $\mathcal{A}^{(0)} = (\Sigma, X)$ and $\mathcal{B}^{(0)} = (\Lambda, Y)$ be zeroth-order rewriting systems. A zeroth-order morphism from $\mathcal{A}^{(0)}$ to $\mathcal{B}^{(0)}$ is an ordered pair $\mathbf{f}^{(0)} = (c, f^{(0)})$, denoted by $\mathbf{f}^{(0)}: \mathcal{A}^{(0)} \longrightarrow \mathcal{B}^{(0)}$, where

- $c = (c_n)_{n \in \mathbb{N}}$ is a derivor from Σ to Λ .
- $f^{(0)}$ is a mapping from X to $T_\Lambda(Y)$.

Following a similar fashion, we define the notions of identity zeroth-order morphism and composition of zeroth-order morphisms.

Definition 5

Let $\mathcal{A}^{(0)} = (\Sigma, X)$, $\mathcal{B}^{(0)} = (\Lambda, Y)$ and $\mathcal{C}^{(0)} = (\Omega, Z)$ be three zeroth-order rewriting systems. Let $\mathbf{f}^{(0)}: \mathcal{A}^{(0)} \longrightarrow \mathcal{B}^{(0)}$ and $\mathbf{g}^{(0)}: \mathcal{B}^{(0)} \longrightarrow \mathcal{C}^{(0)}$ be zeroth-order morphisms where $\mathbf{f}^{(0)} = (c, f^{(0)})$ and $\mathbf{g}^{(0)} = (d, g^{(0)})$. Then $\mathbf{g}^{(0)} \circ \mathbf{f}^{(0)}$, the composition of $\mathbf{f}^{(0)}$ and $\mathbf{g}^{(0)}$, is the zeroth-order morphism $(d \circ c, g^{(0)\sharp} \circ f^{(0)})$, where $d \circ c$ is the composition of derivors, introduced in Definition 3, and $g^{(0)\sharp} \circ f^{(0)}$ is the mapping from X to $T_\Omega(Z)$ being $g^{(0)\sharp}$ the canonical extension of $g^{(0)}$ to the free Λ -algebra $T_\Lambda(Y)$.

For every zeroth-order rewriting system $\mathcal{A}^{(0)} = (\Sigma, X)$, the identity zeroth-order morphism at (Σ, X) is (η^Σ, η^X) .

Finally, we show that they constitute a category.

Proposition 5

[9, Proposition 2.11.5] Zeroth-order rewriting systems and zeroth-order morphisms constitute a category, that we denote by $\text{Rws}_\delta^{(0)}$.

First-order morphisms

The generalization to rewriting systems is based on the study of rewriting systems done in parts one and two of [9]. Let us begin by defining the notion of morphism between rewriting systems.

Definition 6

Let $\mathcal{A} = (\Sigma, X, \mathcal{A})$ and $\mathcal{B} = (\Lambda, Y, \mathcal{B})$ be rewriting systems. A morphism from $\mathcal{A}$ to $\mathcal{B}$ is an ordered pair $\mathbf{f}^{(1)} = (c, (f^{(i)})_{i \in \mathbb{N}})$, denoted by $\mathbf{f}^{(1)} : \mathcal{A} \longrightarrow \mathcal{B}$, where

1. $\mathbf{f}^{(0)} = (c, f^{(0)})$, the underlying zeroth-order morphism of $\mathbf{f}^{(1)}$, is a zeroth-order morphism from (Σ, X) to (Λ, Y) , introduced in Definition 4; and
2. $f^{(1)} : \mathcal{A} \longrightarrow \text{Pth}_{\mathcal{B}}$ is a mapping satisfying that, for every rewrite rule $\mathbf{p} = (M, N) \in \mathcal{A}$, we have that

$$f^{(1)}(\mathbf{p}) \in \text{Pth}_{\mathcal{B}}(f^{(0)\#}(M), f^{(0)\#}(N)).$$

The alternative notation $\mathbf{f}^{(1)} = (\mathbf{f}^{(0)}, f^{(1)})$ will also be used.

We now show that, given $\mathbf{f}^{(1)} = (\mathbf{f}^{(0)}, f^{(1)})$ a morphism of rewriting systems, we can extend the mapping $f^{(1)}$ to the set of paths in $\mathcal{A}$, i.e., $\text{Pth}_{\mathcal{A}}$.

Proposition 6

[9, Proposition 34.1.1] Let $\mathbf{f}^{(1)} = (c, (f^{(i)})_{i \in \mathbb{N}}) : \mathcal{A} \longrightarrow \mathcal{B}$ be a morphism. Then there exists a mapping $f^{(1)\flat}$ from $\text{Pth}_{\mathcal{A}}$ to $\text{Pth}_{\mathcal{B}}$, which we call the path extension mapping of $f^{(1)}$, satisfying that

1. $\text{sc}_{\mathcal{B}}^{(0,1)} \circ f^{(1)\flat} = f^{(0)\#} \circ \text{sc}_{\mathcal{A}}^{(0,1)}$.
2. $\text{tg}_{\mathcal{B}}^{(0,1)} \circ f^{(1)\flat} = f^{(0)\#} \circ \text{tg}_{\mathcal{A}}^{(0,1)}$.
3. $f^{(1)\flat} \circ \text{ip}_{\mathcal{A}}^{(1,0)\#} = \text{ip}_{\mathcal{B}}^{(1,0)\#} \circ f^{(0)\#}$;
4. $f^{(1)\flat} \circ \text{ech}_{\mathcal{A}}^{(1,\mathcal{A})} = f^{(1)}$.

Proof. Let us define $f^{(1)\flat}$ by Artinian recursion on $(\text{Pth}_{\mathcal{A}}, \leq_{\text{Pth}_{\mathcal{A}}})$ as follows.

Base step of the Artinian recursion.

Let $\mathfrak{P}$ be a minimal element of $(\text{Pth}_{\mathcal{A}}, \leq_{\text{Pth}_{\mathcal{A}}})$. Then $\mathfrak{P}$ is either (1) an $(1, 0)$ -identity path or (2) an echelon. If (1), then $\mathfrak{P} = \text{ip}_{\mathcal{A}}^{(1,0)\#}(P)$ for some term $P \in T_{\Sigma}(X)$. We define $f^{(1)\flat}(\mathfrak{P})$ to be the $(1, 0)$ -identity path at $f^{(0)\#}(P)$ which is a term in $T_{\Lambda}(Y)$, i.e., $f^{(1)\flat}(\mathfrak{P}) = \text{ip}_{\mathcal{B}}^{(1,0)\#}(f^{(0)\#}(P))$. If (2), i.e., if $\mathfrak{P}$ is an echelon associated with $\mathbf{p} \in \mathcal{A}$ then we define $f^{(1)\flat}(\mathfrak{P}) = f^{(1)}(\mathbf{p})$.

Inductive step of the Artinian recursion.

Let $\mathfrak{P}$ be a non-minimal element of $(\text{Pth}_{\mathcal{A}}, \leq_{\text{Pth}_{\mathcal{A}}})$. We can assume that $\mathfrak{P}$ is not a $(1, 0)$ -identity path, since those paths already have an image for the path extension mapping. Let us suppose that, for every path $\mathfrak{Q} \in \text{Pth}_{\mathcal{A}}$, if $\mathfrak{Q} <_{\text{Pth}_{\mathcal{A}}} \mathfrak{P}$, then the value of the path extension mapping at $\mathfrak{Q}$ has already been defined.

We have that $\mathfrak{P}$ is either (1) a path of length m strictly greater than one containing at least one echelon or (2) an echelonless path.

If (1), let $i \in m$ be the first index for which the one-step subpath $\mathfrak{P}^{i,i}$ of $\mathfrak{P}$ is an echelon. We consider different cases for i according to the definition of $\prec_{\text{Pth}_{\mathcal{A}}}$.

If $i = 0$, we have that the paths $\mathfrak{P}^{0,0}$ and $\mathfrak{P}^{1,m-1} \prec_{\text{Pth}_{\mathcal{A}}}$ -precede the path $\mathfrak{P}$. In this case, we set $f^{(1)b}(\mathfrak{P}) = f^{(1)b}(\mathfrak{P}^{1,m-1}) \circ^{0\text{Pth}_{\mathcal{B}}} f^{(1)b}(\mathfrak{P}^{0,0})$.

If $i \neq 0$, we have that the paths $\mathfrak{P}^{0,i-1}$ and $\mathfrak{P}^{i,m-1} \prec_{\text{Pth}_{\mathcal{A}}}$ -precede the path $\mathfrak{P}$. In this case, we set $f^{(1)b}(\mathfrak{P}) = f^{(1)b}(\mathfrak{P}^{i,m-1}) \circ^{0\text{Pth}_{\mathcal{B}}} f^{(1)b}(\mathfrak{P}^{0,i-1})$.

If (2), i.e., if $\mathfrak{P}$ is an echelonless path in $\text{Pth}_{\mathcal{A},s}$, then the conditions for the path extraction algorithm are met. Then, there exists a unique n -ary operation symbol $\sigma \in \Sigma_n$ associated to $\mathfrak{P}$. Let $(\mathfrak{P}_j)_{j \in n}$ be the family of paths in $\text{Pth}_{\mathcal{A}}^n$ which we can extract from $\mathfrak{P}$. In this case, we set $f^{(1)b}(\mathfrak{P}) = \sigma^{\text{Pth}_{\mathcal{B}}}((f^{(1)b}(\mathfrak{P}_j))_{j \in n})$. $\square$

Proposition 7

[9, Propositions 35.1.1, 35.1.3, 35.2.1, 35.2.2 and 35.3.1] Let $\mathbf{f}^{(1)} = (c, (f^{(i)})_{i \in 2})$ be a morphism from $\mathcal{A}$ to $\mathcal{B}$. Then the sets $\text{Pth}_{\mathcal{B}}$ and $[\text{Pth}_{\mathcal{B}}]$ are equipped, in a natural way, with a structure of partial $\Sigma^{\mathcal{A}}$ -algebra that we denote by $\mathbf{Pth}_{\mathcal{B}}^{\mathbf{f}^{(1)}}$ and $[\mathbf{Pth}_{\mathcal{B}}^{\mathbf{f}^{(1)}}]$, respectively. The sets $\text{PT}_{\mathcal{B}}$ and $[\text{PT}_{\mathcal{B}}]$ are equipped, in a natural way, with a structure of partial $\Sigma^{\mathcal{A}}$ -algebra that we denote by $\mathbf{PT}_{\mathcal{B}}^{\mathbf{f}^{(1)}}$ and $[\mathbf{PT}_{\mathcal{B}}^{\mathbf{f}^{(1)}}]$, respectively. The set $\mathbf{T}_{\mathcal{B}}(\mathbf{Pth}_{\mathcal{B}})$ is equipped, in a natural way, with a structure of partial $\Sigma^{\mathcal{A}}$ -algebra that we denote by $\mathbf{T}_{\mathcal{B}}^{\mathbf{f}^{(1)}}(\mathbf{Pth}_{\mathcal{B}})$.

Proposition 8

[9, Theorems 35.2.6 and 35.3.4] The partial $\Sigma^{\mathcal{A}}$ -algebras $[\mathbf{Pth}_{\mathcal{B}}^{\mathbf{f}^{(1)}}]$, $[\mathbf{PT}_{\mathcal{B}}^{\mathbf{f}^{(1)}}]$ and $\mathbf{T}_{\mathcal{B}}^{\mathbf{f}^{(1)}}(\mathbf{Pth}_{\mathcal{B}})$ are isomorphic and belong to $\text{PAlg}(\mathcal{E}^{\mathcal{A}})$.

Quotient path-extension mapping

In order to complete the study of morphisms between rewriting systems, we show that the path-extension mapping is compatible with the kernels of the Curry-Howard mappings of $\mathcal{A}$ and $\mathcal{B}$. Thus defining a mapping from $[\text{Pth}_{\mathcal{A}}]$ to $[\text{Pth}_{\mathcal{B}}]$.

Proposition 9

[9, Proposition 36.0.2] The mapping $\text{pr}_{\mathcal{B}}^{\text{Ker}(\text{CH}^{(1)})} \circ f^{(1)b}$, from $\text{Pth}_{\mathcal{A}}$ to $[\text{Pth}_{\mathcal{B}}]$, is a $\Sigma^{\mathcal{A}}$ -homomorphism from $\mathbf{Pth}_{\mathcal{A}}$ to $[\mathbf{Pth}_{\mathcal{B}}^{\mathbf{f}^{(1)}}]$ satisfying that

$$\text{Ker}(\text{CH}_{\mathcal{A}}^{(1)}) \subseteq \text{Ker}(\text{pr}_{\mathcal{B}}^{\text{Ker}(\text{CH}^{(1)})} \circ f^{(1)b}).$$

Definition 7

Following Proposition 9 and taking into account the Universal Property of the Quotient, there exists a unique $\Sigma^{\mathcal{A}}$ -homomorphism, that we will denote by $f^{[1]@}$, i.e., $f^{[1]@}: [\mathbf{Pth}_{\mathcal{A}}] \rightarrow$

$[\mathbf{Pth}_{\mathcal{B}}^{f^{(1)}}]$, satisfying that

$$f^{[1]@} \circ \text{pr}_{\mathcal{A}}^{\text{Ker}(\text{CH}^{(1)})} = \text{pr}_{\mathcal{B}}^{\text{Ker}(\text{CH}^{(1)})} \circ f^{(1)b},$$

namely $f^{[1]@} = (\text{pr}_{\mathcal{B}}^{\text{Ker}(\text{CH}^{(1)})} \circ f^{(1)b})^{\sharp}$. We will call this mapping the quotient path extension mapping of $f^{(1)}$. Formally, for every path class $[\mathfrak{P}]$ in $[\mathbf{Pth}_{\mathcal{A}}]$, $f^{[1]@}([\mathfrak{P}]) = [f^{(1)b}(\mathfrak{P})]$.

Remark. Let us recall that the partial $\Sigma^{\mathcal{A}}$ -algebras $[\mathbf{Pth}_{\mathcal{A}}]$ and $\mathbf{T}_{\mathcal{E}^{\mathcal{A}}}(\mathbf{Pth}_{\mathcal{A}})$, which is the free partial $\Sigma^{\mathcal{A}}$ -algebra in the category $\mathbf{PAlg}(\mathcal{E}^{\mathcal{A}})$ determined by $\mathbf{Pth}_{\mathcal{A}}$, are isomorphic. Thus, the construction of the quotient path-extension mapping could be done taking into account that the partial $\Sigma^{\mathcal{A}}$ -algebra $[\mathbf{Pth}_{\mathcal{B}}^{f^{(1)}}]$ belongs to $\mathbf{PAlg}(\mathcal{E}^{\mathcal{A}})$ and use the universal property of $\mathbf{T}_{\mathcal{E}^{\mathcal{A}}}(\mathbf{Pth}_{\mathcal{A}})$. Both constructions define the same mapping.

We now present the different relations of the quotient path-extension mapping with several defined mappings.

Proposition 10

[9, Propositions 36.1.1, 36.1.2, 36.1.3 and 36.1.4] Let $f^{(1)} = (c, (f^{(i)})_{i \in 2})$ be a morphism from $\mathcal{A}$ to $\mathcal{B}$. Then the following equalities hold

- $\text{CH}_{\mathcal{B}}^{[1]} \circ f^{[1]@} = f^{[1]@} \circ \text{CH}_{\mathcal{A}}^{[1]}$.
- $\text{ip}_{\mathcal{B}}^{([1], Y)^@} \circ f^{[1]@} = f^{[1]@} \circ \text{ip}_{\mathcal{A}}^{([1], X)^@}$.
- $\text{sc}_{\mathcal{B}}^{(0, [1])} \circ f^{[1]@} = f^{(0)\sharp} \circ \text{sc}_{\mathcal{A}}^{(0, [1])}$.
- $\text{tg}_{\mathcal{B}}^{(0, [1])} \circ f^{[1]@} = f^{(0)\sharp} \circ \text{tg}_{\mathcal{A}}^{(0, [1])}$.
- $\text{ip}_{\mathcal{B}}^{([1], 0)^{\sharp}} \circ f^{(0)\sharp} = f^{[1]@} \circ \text{ip}_{\mathcal{A}}^{([1], 0)^{\sharp}}$.

The category $\mathbf{Rws}_{\mathfrak{d}}^{[1]}$

In order to define a category, we begin by defining the notion of identity morphism and that of composition of morphisms.

Definition 8

Let $f^{(1)}: \mathcal{A} \rightarrow \mathcal{B}$ and $g^{(1)}: \mathcal{B} \rightarrow \mathcal{C}$ be morphisms. Then $g^{(1)} \circ f^{(1)}$, the composition of $f^{(1)}$ and $g^{(1)}$, is the morphism $(g^{(0)} \circ f^{(0)}, g^{(1)b} \circ f^{(1)})$, where $g^{(0)} \circ f^{(0)}$ is the composition of the underlying zeroth-order morphisms, introduced in Definition 5, and $g^{(1)b} \circ f^{(1)}$ is the mapping from $\mathcal{A}$ to $\mathbf{Pth}_{\mathcal{C}}$ being $g^{(1)b}$ the path extension mapping of $g^{(1)}$ introduced in Proposition 6.

Let $\mathcal{A} = (\Sigma, X, \mathcal{A})$ be a rewriting system. The identity morphism at $\mathcal{A}$ is given by $(\eta^{\Sigma}, \eta^X, \text{ech}_{\mathcal{A}}^{(1, \mathcal{A})})$, where $\text{ech}_{\mathcal{A}}^{(1, \mathcal{A})}$ is the echelon mapping associated with the rewriting system $\mathcal{A}$.

Proposition 11

[9, Propositions 37.1.3 and 37.1.9] Let $f^{(1)}: \mathcal{A} \rightarrow \mathcal{B}$ and $g^{(1)}: \mathcal{B} \rightarrow \mathcal{C}$ be morphisms. Thus,

1. $\text{ech}_{\mathcal{A}}^{(1, \mathcal{A})@} = \text{id}^{[\text{Pth}_{\mathcal{A}}]}$.
2. $(g^{(1)b} \circ f^{(1)})^@ = g^{[1]@} \circ f^{[1]@}$.

Rewriting systems and morphisms between them do not form a category since, among other things, the composition of morphisms is not associative. To overcome this problem, we introduce an equivalence relation on the morphisms.

Definition 9

Let $\mathbf{f}^{(1)}, \mathbf{g}^{(1)}: \mathcal{A} \rightarrow \mathcal{B}$ be morphisms. We will say that $\mathbf{f}^{(1)}$ and $\mathbf{g}^{(1)}$ are equivalent, written $\mathbf{f}^{(1)} \cong^{(1)} \mathbf{g}^{(1)}$, if

1. $(c, f^{(0)}) = (d, g^{(0)})$, i.e., $c = d$ and $f^{(0)} = g^{(0)}$; and
2. $\text{pr}_{\mathcal{B}}^{\text{Ker}(\text{CH}^{(1)})} \circ f^{(1)b} = \text{pr}_{\mathcal{B}}^{\text{Ker}(\text{CH}^{(1)})} \circ g^{(1)b}$. That is, for every path $\mathfrak{P}$ in $\text{Pth}_{\mathcal{A}}$ $[f^{(1)b}(\mathfrak{P})] = [g^{(1)b}(\mathfrak{P})]$.

Note that $\cong^{(1)}$ is an equivalence relation. Therefore, to simplify notation, we will denote by $[\mathbf{f}^{(1)}]$ the equivalence class $[\mathbf{f}^{(1)}]_{\cong^{(1)}}$.

We now show that the composition of morphism classes does not depend on the representative of the $\cong^{(1)}$ -classes.

Proposition 12

[9, Proposition 37.1.10] Let $\mathbf{f}^{(1)}, \mathbf{f}'^{(1)}: \mathcal{A} \rightarrow \mathcal{B}$ and $\mathbf{g}^{(1)}, \mathbf{g}'^{(1)}: \mathcal{B} \rightarrow \mathcal{C}$ be morphisms. If $\mathbf{f}^{(1)} \cong^{(1)} \mathbf{f}'^{(1)}$ and $\mathbf{g}^{(1)} \cong^{(1)} \mathbf{g}'^{(1)}$, then $\mathbf{g}^{(1)} \circ \mathbf{f}^{(1)} \cong^{(1)} \mathbf{g}'^{(1)} \circ \mathbf{f}'^{(1)}$. Thus, the composition of their equivalent classes $[\mathbf{g}^{(1)}] \circ [\mathbf{f}^{(1)}] = [\mathbf{g}^{(1)} \circ \mathbf{f}^{(1)}]$ is well-defined and does not depend on the representatives.

Proposition 13

[9, Proposition 37.1.11] Rewriting systems together with the equivalence classes of morphisms constitute a category, that we denote by $\text{Rws}_0^{[1]}$.

Proposition 14

[9, Proposition 37.2.1] Let $\mathbf{f}^{(1)}, \mathbf{g}^{(1)}: \mathcal{A} \rightarrow \mathcal{B}$ be morphisms such that $\mathbf{f}^{(1)} \cong^{(1)} \mathbf{g}^{(1)}$. Then $[\text{Pth}_{\mathcal{B}}^{\mathbf{f}^{(1)}}] = [\text{Pth}_{\mathcal{B}}^{\mathbf{g}^{(1)}}]$ and $f^{[1]@} = g^{[1]@}$.

We next introduce the notion of tower and morphisms between them.

Definition 10

The tower associated to the rewriting system $\mathcal{A}$ is the ordered pair $\mathbb{A} = (\mathcal{A}, \text{T}_{\mathcal{E}\mathcal{A}}(\text{Pth}_{\mathcal{A}}))$.

A tower morphism from $\mathbb{A} = (\mathcal{A}, \text{T}_{\mathcal{E}\mathcal{A}}(\text{Pth}_{\mathcal{A}}))$ to $\mathbb{B} = (\mathcal{B}, \text{T}_{\mathcal{E}\mathcal{B}}(\text{Pth}_{\mathcal{B}}))$ is an ordered pair $\mathbf{f}^{[1]@} = ([\mathbf{f}^{(1)}], f^{[1]@})$, denoted by $\mathbf{f}^{[1]@}: \mathbb{A} \rightarrow \mathbb{B}$, where $[\mathbf{f}^{(1)}]$ is the equivalence class of a morphism $\mathbf{f}^{(1)}: \mathcal{A} \rightarrow \mathcal{B}$ and $f^{[1]@}$ is its quotient path extension mapping.

Definition 11

Let $\mathbf{f}^{[1]@}: \mathbb{A} \rightarrow \mathbb{B}$ and $\mathbf{g}^{[1]@}: \mathbb{B} \rightarrow \mathbb{C}$ be tower morphisms. Then $\mathbf{g}^{[1]@} \circ \mathbf{f}^{[1]@}$, the composition of $\mathbf{f}^{[1]@}$ and $\mathbf{g}^{[1]@}$ is the tower morphism $([\mathbf{g}^{(1)}] \circ [\mathbf{f}^{(1)}], g^{[1]@} \circ f^{[1]@})$, where, according to Proposition 12 the composition $[\mathbf{g}^{(1)}] \circ [\mathbf{f}^{(1)}] = [\mathbf{g}^{(1)} \circ \mathbf{f}^{(1)}]$ is well-defined and, according to Proposition 11, $g^{[1]@} \circ f^{[1]@}$ is its quotient path extension mapping.

Let $\mathbb{A} = (\mathcal{A}, \mathbf{T}_{\mathcal{E}\mathcal{A}}(\mathbf{Pth}_{\mathcal{A}}))$ be a tower. The identity tower morphism at $\mathbb{A}$ is the ordered pair $([\text{id}^{\mathcal{A}}], \text{id}^{\mathbf{T}_{\mathcal{E}\mathcal{A}}(\mathbf{Pth}_{\mathcal{A}})})$ where, $[\text{id}^{\mathcal{A}}]$ is the equivalence class of the identity morphism at $\mathcal{A}$, introduced in Definition 8, and, according to Proposition 11, $\text{id}^{\mathbf{T}_{\mathcal{E}\mathcal{A}}(\mathbf{Pth}_{\mathcal{A}})}$ is its quotient path extension mapping.

Proposition 15

[9, Proposition 37.2.5] Towers together with tower morphisms constitute a category, that we denote by $\text{Tw}_{\mathfrak{D}}^{[1]}$.

Proposition 16

[9, Proposition 37.2.10] The categories $\text{Rws}_{\mathfrak{D}}^{[1]}$ and $\text{Tw}_{\mathfrak{D}}^{[1]}$ are isomorphic.

Second-order morphisms

Let us begin by defining the notion of second-order morphism between second-order rewriting systems.

Definition 12

Let $\mathcal{A}^{(2)} = (\mathcal{A}, \mathcal{A}^{(2)})$ and $\mathcal{B}^{(2)} = (\mathcal{B}, \mathcal{B}^{(2)})$ be second-order rewriting systems. A second-order morphism from $\mathcal{A}^{(2)}$ to $\mathcal{B}^{(2)}$ is an ordered pair $\mathbf{f}^{(2)} = (c, (f^{(i)})_{i \in \mathbb{N}})$, denoted by $\mathbf{f}^{(2)}: \mathcal{A}^{(2)} \rightarrow \mathcal{B}^{(2)}$, where

1. $\mathbf{f}^{(1)} = (c, (f^{(i)})_{i \in \mathbb{N}})$, the underlying morphism of $\mathbf{f}^{(2)}$, is a morphism from $\mathcal{A}$ to $\mathcal{B}$, introduced in Definition 6; and
2. $f^{(2)}: \mathcal{A}^{(2)} \rightarrow \text{Pth}_{\mathcal{B}^{(2)}}$ is a mapping satisfying that, for every second-order rewrite rule $\mathbf{p}^{(2)} = ([M], [N]) \in \mathcal{A}^{(2)}$, we have that $f^{(2)}(\mathbf{p}^{(2)}) \in \text{Pth}_{\mathcal{B}^{(2)}}(f^{[1]@}([M]), f^{[1]@}([N]))$.

The alternative notation $\mathbf{f}^{(2)} = (\mathbf{f}^{(1)}, f^{(2)})$ will also be used.

We now show that, given $\mathbf{f}^{(2)} = (\mathbf{f}^{(1)}, f^{(2)})$ a second-order morphism of second-order rewriting systems, we can extend the mapping $f^{(2)}$ to the set of second-order paths in $\mathcal{A}^{(2)}$, i.e., $\text{Pth}_{\mathcal{A}^{(2)}}$.

Proposition 17

[9, Proposition 38.1.1] Let $\mathbf{f}^{(2)} = (c, (f^{(i)})_{i \in \mathbb{N}}): \mathcal{A}^{(2)} \rightarrow \mathcal{B}^{(2)}$ be a morphism. Then there exists a mapping $f^{(2)\flat}$ from $\text{Pth}_{\mathcal{A}^{(2)}}$ to $\text{Pth}_{\mathcal{B}^{(2)}}$, which we call the second-order path extension mapping of $f^{(2)}$, satisfying that

1. $\text{sc}_{\mathcal{B}}^{([1],2)} \circ f^{(2)b} = f^{[1]@} \circ \text{sc}_{\mathcal{A}}^{([1],2)}.$
2. $\text{tg}_{\mathcal{B}}^{([1],2)} \circ f^{(2)b} = f^{[1]@} \circ \text{tg}_{\mathcal{A}}^{([1],2)}.$
3. $f^{(2)b} \circ \text{ip}_{\mathcal{A}}^{([2],1)\#} = \text{ip}_{\mathcal{B}}^{(2,[1])\#} \circ f^{[1]@};$
4. $f^{(2)b} \circ \text{ech}_{\mathcal{A}^{(2)}}^{(2,\mathcal{A}^{(2)})} = f^{(2)}.$

Proof. Let us define $f^{(2)b}$ by Artinian recursion on $(\text{Pth}_{\mathcal{A}^{(2)}}, \leq_{\text{Pth}_{\mathcal{A}^{(2)}}})$ as follows.

Base step of the Artinian recursion.

Let $\mathfrak{P}^{(2)}$ be a minimal element of $(\text{Pth}_{\mathcal{A}^{(2)}}, \leq_{\text{Pth}_{\mathcal{A}^{(2)}}})$. Then $\mathfrak{P}^{(2)}$ is either (1) an $(2, [1])$ -identity second-order path or (2) a second-order echelon.

If (1), then $\mathfrak{P}^{(2)} = \text{ip}^{(2,[1])\#}([P])$ for some path term class $[P]$ in $[\text{PT}_{\mathcal{A}}]$. we define $f^{(2)b}(\mathfrak{P}^{(2)})$ to be the $(2, [1])$ -identity path at $f^{[1]@}([P])$ which is a path term class in $[\text{PT}_{\mathcal{B}}]$, i.e., $f^{(2)b}(\mathfrak{P}^{(2)}) = \text{ip}_{\mathcal{B}^{(2)}}^{(2,[1])\#}(f^{[1]@}([P]))$.

If (2), if $\mathfrak{P}^{(2)}$ is a second-order echelon associated to a second-order rewrite rule $\mathfrak{p}^{(2)}$ then we define $f^{(2)b}(\mathfrak{P}^{(2)}) = f^{(2)}(\mathfrak{p})$.

Inductive step of the Artinian recursion.

Let $\mathfrak{P}^{(2)}$ be a non-minimal element of $(\text{Pth}_{\mathcal{A}^{(2)}}, \leq_{\text{Pth}_{\mathcal{A}^{(2)}}})$. We can assume that $\mathfrak{P}^{(2)}$ is not a $(2, [1])$ -identity second-order path, since those second-order paths already have an image for the second-order path extension mapping. Let us suppose that, for every second-order path $\Omega^{(2)} \in \text{Pth}_{\mathcal{A}^{(2)}}$, if $\Omega^{(2)} <_{\text{Pth}_{\mathcal{A}^{(2)}}} \mathfrak{P}^{(2)}$, then the value of the second-order path extension mapping at $\Omega^{(2)}$ has already been defined.

We have that $\mathfrak{P}^{(2)}$ is either (1) a second-order path of length m strictly greater than one containing at least one second-order echelon or (2) an echelonless second-order m -path.

If (1), let $i \in m$ be the first index for which the one-step subpath $\mathfrak{P}^{(2),i,i}$ of $\mathfrak{P}^{(2)}$ is a second-order echelon. We consider different cases for i according to the definition of $\prec_{\text{Pth}_{\mathcal{A}^{(2)}}}$.

If $i = 0$, we have that the second-order paths $\mathfrak{P}^{(2),0,0}$ and $\mathfrak{P}^{(2),1,m-1} \prec_{\text{Pth}_{\mathcal{A}^{(2)}}}$ -precede the second-order path $\mathfrak{P}^{(2)}$. In this case, we set

$$f^{(2)b}(\mathfrak{P}^{(2)}) = f^{(2)b}(\mathfrak{P}^{(2),1,m-1}) \circ^{1\text{Pth}_{\mathcal{B}^{(2)}}} f^{(2)b}(\mathfrak{P}^{(2),0,0}).$$

If $i \neq 0$, we have that the second-order paths $\mathfrak{P}^{(2),0,i-1}$ and $\mathfrak{P}^{(2),i,m-1} \prec_{\text{Pth}_{\mathcal{A}^{(2)}}}$ -precede the second-order path $\mathfrak{P}^{(2)}$. In this case, we set

$$f^{(2)b}(\mathfrak{P}^{(2)}) = f^{(2)b}(\mathfrak{P}^{(2),i,m-1}) \circ^{1\text{Pth}_{\mathcal{B}^{(2)}}} f^{(2)b}(\mathfrak{P}^{(2),0,i-1}).$$

If (2), i.e., if $\mathfrak{P}^{(2)}$ is an echelonless second-order path in $\text{Pth}_{\mathcal{A}^{(2)}}$. It could be the case that (2.1) $\mathfrak{P}^{(2)}$ is not head-constant. Then let $i \in m$ be the maximum index for which the subpath $\mathfrak{P}^{(2),0,i}$ of $\mathfrak{P}^{(2)}$ is a head-constant, echelonless second-order path. Note that

the second-order pairs $\mathfrak{P}^{(2),0,i}$ and $\mathfrak{P}^{(2),i+1,m-1} \prec_{\text{Pth}_{\mathcal{A}^{(2)}}}$ precede the second-order path $\mathfrak{P}^{(2)}$. In this case, we set

$$f^{(2)b}(\mathfrak{P}^{(2)}) = f^{(2)b}(\mathfrak{P}^{(2),i+1,m-1}) \circ^{1\text{Pth}_{\mathcal{B}^{(2)}}} f^{(2)b}(\mathfrak{P}^{(2),0,i}).$$

Therefore, we are left with the case of $\mathfrak{P}^{(2)}$ being a head-constant echelonless second-order path. It could be the case that (2.2) $\mathfrak{P}^{(2)}$ is not coherent. Then let $i \in m$ be the maximum index for which the subpath $\mathfrak{P}^{(2),0,i}$ of $\mathfrak{P}^{(2)}$ is a coherent head-constant echelonless second-order path. Note that the pairs $\mathfrak{P}^{(2),0,i}$ and $\mathfrak{P}^{(2),i+1,m-1} \prec_{\text{Pth}_{\mathcal{A}^{(2)}}}$ precede the second-order path $\mathfrak{P}^{(2)}$. We set

$$f^{(2)b}(\mathfrak{P}^{(2)}) = f^{(2)b}(\mathfrak{P}^{(2),i+1,m-1}) \circ^{1\text{Pth}_{\mathcal{B}^{(2)}}} f^{(2)b}(\mathfrak{P}^{(2),0,i}).$$

Therefore, we are left with the case (2.3) of $\mathfrak{P}^{(2)}$ being a coherent head-constant echelonless second-order path. In this setting, the conditions for the second-order extraction algorithm are met. Then there exist a unique $n \in \mathbb{N}$ and a unique n -ary operation symbol $\tau \in \Sigma_n^{\mathcal{A}}$ associated with $\mathfrak{P}^{(2)}$. Let $(\mathfrak{P}_j^{(2)})_{j \in n}$ be the family of second-order paths in $\text{Pth}_{\mathcal{A}^{(2)}}^n$ which we can extract from $\mathfrak{P}^{(2)}$. It could be the case that (2.3.1) $\tau = \sigma$, for some $\sigma \in \Sigma$. In this case, we set

$$f^{(2)b}(\mathfrak{P}^{(2)}) = \sigma^{c_{\mathfrak{P}^{(2)}}^*(\text{Pth}_{\mathcal{B}^{(2)}}^{(0,2)})}((f^{(2)b}(\mathfrak{P}_j^{(2)}))_{j \in n}).$$

Therefore, we are left with the case (2.3.2) of τ being the operation symbol $\circ^0$. In this case, we set

$$f^{(2)b}(\mathfrak{P}^{(2)}) = f^{(2)b}(\mathfrak{P}_1^{(2)}) \circ^{0\text{Pth}_{\mathcal{B}^{(2)}}} f^{(2)b}(\mathfrak{P}_0^{(2)}).$$

□

Proposition 18

[9, Propositions 39.1.1, 39.1.3, 39.2.1, 39.2.2 and 39.3.1] Let $\mathbf{f}^{(2)} = (c, (f^{(i)})_{i \in 3})$ be a morphism from $\mathcal{A}^{(2)}$ to $\mathcal{B}^{(2)}$. Then the sets $\text{Pth}_{\mathcal{B}^{(2)}}$ and $\llbracket \text{Pth}_{\mathcal{B}^{(2)}} \rrbracket$ are equipped, in a natural way, with a structure of partial $\Sigma^{\mathcal{A}^{(2)}}$ -algebra that we denote by $\text{Pth}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}}$ and $\llbracket \text{Pth}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}} \rrbracket$, respectively. The sets $\text{PT}_{\mathcal{B}^{(2)}}$ and $\llbracket \text{PT}_{\mathcal{B}^{(2)}} \rrbracket$ are equipped, in a natural way, with a structure of partial $\Sigma^{\mathcal{A}^{(2)}}$ -algebra that we denote by $\text{PT}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}}$ and $\llbracket \text{PT}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}} \rrbracket$, respectively. The set $\text{T}_{\mathcal{E}^{\mathcal{B}^{(2)}}}(\text{Pth}_{\mathcal{B}^{(2)}})$ is equipped, in a natural way, with a structure of partial $\Sigma^{\mathcal{A}^{(2)}}$ -algebra that we denote by $\text{T}_{\mathcal{E}^{\mathcal{B}^{(2)}}}^{\mathbf{f}^{(2)}}(\text{Pth}_{\mathcal{B}^{(2)}})$.

Proof. By Proposition 4. □

Proposition 19

[9, Theorems 39.2.6 and 39.3.4] The partial $\Sigma^{\mathcal{A}^{(2)}}$ -algebras $\llbracket \text{Pth}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}} \rrbracket$, $\llbracket \text{PT}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}} \rrbracket$ and $\text{T}_{\mathcal{E}^{\mathcal{B}^{(2)}}}^{\mathbf{f}^{(2)}}(\text{Pth}_{\mathcal{B}^{(2)}})$ are isomorphic and belong to $\text{PAlg}(\mathcal{E}^{\mathcal{A}^{(2)}})$.

Second-order quotient path extension mapping

In order to complete the study of second-order morphisms between second-order rewriting systems, we show that the second-order path-extension mapping is compatible with the supremum of the kernels of the second-order Curry-Howard mappings and the congruences $\Upsilon^{[1]}$ of $\mathcal{A}^{(2)}$ and $\mathcal{B}^{(2)}$. Thus defining a mapping from $\llbracket \text{Pth}_{\mathcal{A}^{(2)}} \rrbracket$ to $\llbracket \text{Pth}_{\mathcal{B}^{(2)}} \rrbracket$.

Proposition 20

[9, Proposition 40.0.3] The mapping $\text{pr}_{\mathcal{B}^{(2)}}^{\llbracket \cdot \rrbracket} \circ f^{(2)\flat}$ from $\text{Pth}_{\mathcal{A}^{(2)}}$ to $\llbracket \text{Pth}_{\mathcal{B}^{(2)}} \rrbracket$ is a $\Sigma^{\mathcal{A}^{(2)}}$ -homomorphism from $\text{Pth}_{\mathcal{A}^{(2)}}$ to $\llbracket \text{Pth}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}} \rrbracket$ satisfying that

$$\text{Ker}(\text{CH}_{\mathcal{A}^{(2)}}^{(2)}) \vee \Upsilon^{[1]} \subseteq \text{Ker}(\text{pr}_{\mathcal{B}^{(2)}}^{\llbracket \cdot \rrbracket} \circ f^{(2)\flat}).$$

Definition 13

Following Proposition 20 and taking into account the Universal Property of the Quotient, there exists a unique $\Sigma^{\mathcal{A}^{(2)}}$ -homomorphism, that we will denote by $f^{\llbracket 2 \rrbracket @}$, i.e., $f^{\llbracket 2 \rrbracket @} : \llbracket \text{Pth}_{\mathcal{A}^{(2)}} \rrbracket \longrightarrow \llbracket \text{Pth}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}} \rrbracket$, satisfying that

$$f^{\llbracket 2 \rrbracket @} \circ \text{pr}_{\mathcal{A}^{(2)}}^{\llbracket \cdot \rrbracket} = \text{pr}_{\mathcal{B}^{(2)}}^{\llbracket \cdot \rrbracket} \circ f^{(2)\flat},$$

namely $f^{\llbracket 2 \rrbracket @} = (\text{pr}_{\mathcal{B}^{(2)}}^{\llbracket \cdot \rrbracket} \circ f^{(2)\flat})^{\sharp}$. We will call this mapping the second-order quotient path extension mapping of $\mathbf{f}^{(2)}$. Formally, for every second-order path class $\llbracket \mathfrak{P}^{(2)} \rrbracket$ in $\llbracket \text{Pth}_{\mathcal{A}^{(2)}} \rrbracket$, $f^{\llbracket 2 \rrbracket @}(\llbracket \mathfrak{P}^{(2)} \rrbracket) = \llbracket f^{(2)\flat}(\mathfrak{P}^{(2)}) \rrbracket$.

Remark. Let us recall that the partial $\Sigma^{\mathcal{A}^{(2)}}$ -algebras $\llbracket \text{Pth}_{\mathcal{A}^{(2)}} \rrbracket$ and $\mathbf{T}_{\mathcal{E}^{\mathcal{A}^{(2)}}}(\text{Pth}_{\mathcal{A}^{(2)}})$, which is the free partial $\Sigma^{\mathcal{A}^{(2)}}$ -algebra in the category $\mathbf{PAlg}(\mathcal{E}^{\mathcal{A}^{(2)}})$ determined by $\text{Pth}_{\mathcal{A}^{(2)}}$, are isomorphic. Thus, the construction of the quotient path-extension mapping could be done taking into account that the partial $\Sigma^{\mathcal{A}^{(2)}}$ -algebra $\llbracket \text{Pth}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}} \rrbracket$ belongs to $\mathbf{PAlg}(\mathcal{E}^{\mathcal{A}^{(2)}})$ and use the universal property of $\mathbf{T}_{\mathcal{E}^{\mathcal{A}^{(2)}}}(\text{Pth}_{\mathcal{A}^{(2)}})$. Both constructions define the same mapping.

We now present the different relations of the second-order quotient path-extension mapping with several defined mappings.

Proposition 21

[9, Propositions 40.1.1, 40.1.2, 40.1.3, 40.1.4, 40.1.5 and 40.1.6] Let $\mathbf{f}^{(2)} = (c, (f^{(i)})_{i \in 3})$ be a second-order morphism from $\mathcal{A}^{(2)}$ to $\mathcal{B}^{(2)}$. Then the following equalities hold

- $\text{CH}_{\mathcal{B}}^{\llbracket 2 \rrbracket} \circ f^{\llbracket 2 \rrbracket @} = f^{\llbracket 2 \rrbracket @} \circ \text{CH}_{\mathcal{A}}^{\llbracket 2 \rrbracket}.$
- $\text{ip}_{\mathcal{B}}^{(\llbracket 2 \rrbracket, Y)^@} \circ f^{\llbracket 2 \rrbracket @} = f^{\llbracket 2 \rrbracket @} \circ \text{ip}_{\mathcal{A}}^{(\llbracket 2 \rrbracket, X)^@}.$
- $\text{sc}_{\mathcal{B}}^{([1], \llbracket 2 \rrbracket)} \circ f^{\llbracket 2 \rrbracket @} = f^{\llbracket 1 \rrbracket @} \circ \text{sc}_{\mathcal{A}}^{([1], \llbracket 2 \rrbracket)}.$
- $\text{tg}_{\mathcal{B}}^{([1], \llbracket 2 \rrbracket)} \circ f^{\llbracket 2 \rrbracket @} = f^{\llbracket 1 \rrbracket @} \circ \text{tg}_{\mathcal{A}}^{([1], \llbracket 2 \rrbracket)}.$
- $\text{sc}_{\mathcal{B}}^{(0, \llbracket 2 \rrbracket)} \circ f^{\llbracket 2 \rrbracket @} = f^{(0)\sharp} \circ \text{sc}_{\mathcal{A}}^{(0, \llbracket 2 \rrbracket)}.$
- $\text{tg}_{\mathcal{B}}^{(0, \llbracket 2 \rrbracket)} \circ f^{\llbracket 2 \rrbracket @} = f^{(0)\sharp} \circ \text{tg}_{\mathcal{A}}^{(0, \llbracket 2 \rrbracket)}.$
- $\text{ip}_{\mathcal{B}}^{(\llbracket 2 \rrbracket, [1])\sharp} \circ f^{\llbracket 1 \rrbracket @} = f^{\llbracket 2 \rrbracket @} \circ \text{ip}_{\mathcal{A}}^{(\llbracket 2 \rrbracket, [1])\sharp}.$
- $\text{ip}_{\mathcal{B}}^{(\llbracket 2 \rrbracket, 0)\sharp} \circ f^{(0)\sharp} = f^{\llbracket 2 \rrbracket @} \circ \text{ip}_{\mathcal{A}}^{(\llbracket 2 \rrbracket, 0)\sharp}.$

The category $\mathbf{Rws}_0^{[2]}$

In order to define a category, we begin by defining the notion of identity second-order morphism and that of composition of second-order morphisms.

Definition 14

Let $\mathbf{f}^{(2)} : \mathcal{A}^{(2)} \longrightarrow \mathcal{B}^{(2)}$ and $\mathbf{g}^{(2)} : \mathcal{B}^{(2)} \longrightarrow \mathcal{C}^{(2)}$ be second-order morphisms. Then $\mathbf{g}^{(2)} \circ \mathbf{f}^{(2)}$, the composition of $\mathbf{f}^{(2)}$ and $\mathbf{g}^{(2)}$, is the second-order morphism $(\mathbf{g}^{(1)} \circ \mathbf{f}^{(1)}, g^{(2)\flat} \circ f^{(2)})$, where $\mathbf{g}^{(1)} \circ \mathbf{f}^{(1)}$ is the composition of the underlying first- morphisms, introduced in Definition 8, and $g^{(2)\flat} \circ f^{(2)}$ is the mapping from $\mathcal{A}^{(2)}$ to $\mathbf{Pth}_{\mathcal{C}^{(2)}}$ being $g^{(2)\flat}$ the second-order path extension mapping of $g^{(2)}$ introduced in Proposition 17.

Let $\mathcal{A}^{(2)}$ be a second-order rewriting system. The identity second-order morphism at $\mathcal{A}$ is given by $(\text{id}^{\mathcal{A}}, \text{ech}_{\mathcal{A}^{(2)}}^{(2, \mathcal{A}^{(2)})})$, where $\text{ech}_{\mathcal{A}^{(2)}}^{(2, \mathcal{A}^{(2)})}$ is the second-order echelon mapping associated with the second-order rewriting system $\mathcal{A}^{(2)}$.

Proposition 22

[9, Propositions 41.1.3 and 41.1.9] Let $\mathbf{f}^{(2)} : \mathcal{A}^{(2)} \longrightarrow \mathcal{B}^{(2)}$ and $\mathbf{g}^{(2)} : \mathcal{B}^{(2)} \longrightarrow \mathcal{C}^{(2)}$ be second-order morphisms. Thus,

1. $\text{ech}_{\mathcal{A}^{(2)}}^{(2, \mathcal{A}^{(2)})@} = \text{id}^{[\mathbf{Pth}_{\mathcal{A}^{(2)}}]}$.
2. $(g^{(2)\flat} \circ f^{(2)})@ = g^{[2]@} \circ f^{[2]@}$.

Second-order rewriting systems and morphisms between them do not form a category since, among other things, the composition of morphisms is not associative. To overcome this problem, we introduce an equivalence relation on the morphisms.

Definition 15

Let $\mathbf{f}^{(2)}, \mathbf{g}^{(2)} : \mathcal{A}^{(2)} \longrightarrow \mathcal{B}^{(2)}$ be second-order morphisms. We will say that $\mathbf{f}^{(2)}$ and $\mathbf{g}^{(2)}$ are second-order equivalent, written $\mathbf{f}^{(2)} \cong^{(2)} \mathbf{g}^{(2)}$, if

1. $f^{(1)} \cong^{(1)} g^{(1)}$; and
2. $\text{pr}_{\mathcal{B}^{(2)}}^{[\cdot]} \circ f^{(2)\flat} = \text{pr}_{\mathcal{B}^{(2)}}^{[\cdot]} \circ g^{(2)\flat}$. That is, for every second-order path $\mathfrak{P}^{(2)}$ in $\mathbf{Pth}_{\mathcal{A}^{(2)}}$, $[[f^{(2)\flat}(\mathfrak{P}^{(2)})]] = [[g^{(2)\flat}(\mathfrak{P}^{(2)})]]$.

Note that $\cong^{(2)}$ is an equivalence relation. Therefore, to simplify notation, we will denote by $[[\mathbf{f}^{(2)}]]$ the equivalence class $[[\mathbf{f}^{(2)}]]_{\cong^{(2)}}$.

We now show that the composition of second-order morphisms classes does not depend on the representative of the $\cong^{(2)}$ -classes.

Proposition 23

[9, Proposition 41.1.10] Let $\mathbf{f}^{(2)}, \mathbf{f}'^{(2)} : \mathcal{A}^{(2)} \longrightarrow \mathcal{B}^{(2)}$ and $\mathbf{g}^{(2)}, \mathbf{g}'^{(2)} : \mathcal{B}^{(2)} \longrightarrow \mathcal{C}^{(2)}$ be second-order morphisms. If $\mathbf{f}^{(2)} \cong^{(2)} \mathbf{f}'^{(2)}$ and $\mathbf{g}^{(2)} \cong^{(2)} \mathbf{g}'^{(2)}$, then it holds that $\mathbf{g}^{(2)} \circ \mathbf{f}^{(2)} \cong^{(2)} \mathbf{g}'^{(2)} \circ \mathbf{f}'^{(2)}$. Thus, the composition of their equivalence classes $[[\mathbf{g}^{(2)}]] \circ [[\mathbf{f}^{(2)}]] = [[\mathbf{g}^{(2)} \circ \mathbf{f}^{(2)}]]$ is well-defined and does not depend on the representatives.

Proposition 24

[9, Proposition 41.1.11] Second-order rewriting systems together with the equivalence classes of second-order morphisms constitute a category, that we denote by $\mathbf{Rws}_0^{[2]}$.

Proposition 25

[9, Proposition 41.2.1] Let $\mathbf{f}^{(2)}, \mathbf{g}^{(2)}: \mathcal{A}^{(2)} \rightarrow \mathcal{B}^{(2)}$ be second-order morphisms such that $\mathbf{f}^{(2)} \cong^{(2)} \mathbf{g}^{(2)}$. Then $[\mathbf{Pth}_{\mathcal{B}^{(2)}}^{\mathbf{f}^{(2)}}] = [\mathbf{Pth}_{\mathcal{B}^{(2)}}^{\mathbf{g}^{(2)}}]$ and $f^{[2]@} = g^{[2]@}$.

We next introduce the notion of second-order towers and their morphisms.

Definition 16

The second-order tower associated to the second-order rewriting system $\mathcal{A}^{(2)}$ is $\mathbb{A}^{(2)} = (\mathcal{A}^{(2)}, \mathbf{T}_{\mathcal{E}\mathcal{A}^{(2)}}(\mathbf{Pth}_{\mathcal{A}^{(2)}}))$.

A second-order tower morphism from the second-order tower $\mathbb{A}^{(2)} = (\mathcal{A}^{(2)}, \mathbf{T}_{\mathcal{E}\mathcal{A}^{(2)}}(\mathbf{Pth}_{\mathcal{A}^{(2)}}))$ to the second-order tower $\mathbb{B}^{(2)} = (\mathcal{B}^{(2)}, \mathbf{T}_{\mathcal{E}\mathcal{B}^{(2)}}(\mathbf{Pth}_{\mathcal{B}^{(2)}}))$ is an ordered pair $\mathbf{f}^{[2]@} = ([\mathbf{f}^{(2)}], f^{[2]@})$, denoted by $\mathbf{f}^{[2]@}: \mathbb{A}^{(2)} \rightarrow \mathbb{B}^{(2)}$, where $[\mathbf{f}^{(2)}]$ is the equivalence class of a second-order morphism $\mathbf{f}^{(2)}: \mathcal{A}^{(2)} \rightarrow \mathcal{B}^{(2)}$ and $f^{[2]@}$ is its second-order quotient path extension mapping introduced in Definition 13.

Definition 17

Let $\mathbf{f}^{[2]@}: \mathbb{A}^{(2)} \rightarrow \mathbb{B}^{(2)}$ and $\mathbf{g}^{[2]@}: \mathbb{B}^{(2)} \rightarrow \mathbb{C}^{(2)}$ be second-order tower morphisms. Then, the composition of $\mathbf{f}^{[2]@}$ and $\mathbf{g}^{[2]@}$ is the second-order tower morphism

$$([\mathbf{g}^{(2)}] \circ [\mathbf{f}^{(2)}], g^{[2]@} \circ f^{[2]@}),$$

where, according to Proposition 23, the composition $[\mathbf{g}^{(2)}] \circ [\mathbf{f}^{(2)}] = [\mathbf{g}^{(2)} \circ \mathbf{f}^{(2)}]$ is well-defined and, according to Proposition 22, $g^{[2]@} \circ f^{[2]@}$ is its second-order quotient path extension mapping.

Let $\mathbb{A}^{(2)} = (\mathcal{A}^{(2)}, \mathbf{T}_{\mathcal{E}\mathcal{A}^{(2)}}(\mathbf{Pth}_{\mathcal{A}^{(2)}}))$ be a second-order tower. The identity second-order tower morphism at $\mathbb{A}^{(2)}$ is the ordered pair $([\mathbf{id}^{\mathcal{A}^{(2)}}], \mathbf{id}^{\mathbf{T}_{\mathcal{E}\mathcal{A}^{(2)}}(\mathbf{Pth}_{\mathcal{A}^{(2)}})})$, where, $[\mathbf{id}^{\mathcal{A}^{(2)}}]$ is the equivalence class of the identity second-order morphism at $\mathcal{A}^{(2)}$, introduced in Definition 14, and, according to Proposition 22, $\mathbf{id}^{\mathbf{T}_{\mathcal{E}\mathcal{A}^{(2)}}(\mathbf{Pth}_{\mathcal{A}^{(2)}})}$ is its second-order quotient path extension mapping.

Proposition 26

[9, Proposition 41.2.5] Second-order towers together with second-order tower morphisms constitute a category, that we denote by $\mathbf{Tw}_0^{[2]}$.

Proposition 27

[9, Proposition 41.2.10] The categories $\mathbf{Rws}_0^{[2]}$ and $\mathbf{Tw}_0^{[2]}$ are isomorphic.

Acknowledgements

The second and third authors were supported by the grant PID2024-159495NB-I00 from the Ministerio de Ciencia e Innovación, Spain, and the grant CIAICO/2023/007 from the Conselleria d'Educació, Universitats i Ocupació, Generalitat Valenciana. The second author has held a Specially Appointed Professor position at Nantong University during the completion of this work. The third author was supported by the grant CIACIF/2022/489 from the Conselleria d'Educació, Universitats i Ocupació, Generalitat Valenciana co-funded by the European Social Fund. Special thanks to Asunción de Montesa for her categorical knowledge.

References

- [1] G. M. Bergman. *An invitation to general algebra and universal constructions*, volume 558. Springer, 2015.
- [2] T. S. Blyth. *Lattices and Ordered Algebraic Structures*. Universitext. Springer, London, 2005.
- [3] N. Bourbaki. *Théorie des ensembles*. Hermann, Paris, 1970.
- [4] P. Burmeister. Partial algebras—survey of a unifying approach towards a two-valued model theory for partial algebras. *Algebra Universalis*, 15:306–358, 1982.
- [5] P. Burmeister. A model theoretic oriented approach to partial algebras. i: Introduction to theory and application of partial algebras. *Mathematical Research*, 32, 1986.
- [6] P. Burmeister. Lecture notes on universal algebra. Many-sorted partial algebras. *Manuscript retrieved from <http://www.mathematik.tu-darmstadt.de/~burmeister/>*, 2002.
- [7] S. Burris and H. P. Sankappanavar. *A course in universal algebra*, volume 78. Springer, New York-Berlin, 1981.
- [8] J. Climent Vidal and E. Cosme Llópez. Congruence based proofs of the recognizability theorems for free many-sorted algebras. *Journal of Logic and Computation*, 30:561–633, 2010.
- [9] J. Climent Vidal, E. Cosme Llópez, and R. Ruiz Mora. From higher-order rewriting systems to higher-order categorial algebras and higher-order Curry-Howard isomorphisms, 2026. arXiv:2402.12051.
- [10] J. Climent Vidal and J. Soliveres Tur. A 2-categorical framework for the syntax and semantics of many-sorted equational logic. *Reports on Mathematical Logic*, 45, 2010.
- [11] B. A. Davey and H. A. Priestley. *Introduction to lattices and order*. Cambridge University Press, New York, 2002. Second edition.

- [12] H. B. Enderton. *Elements of set theory*. Academic press, New York-London, 1977.
- [13] J. Goguen, J. Thatcher, and E. Wagner. An initial algebra approach to the specification, correctness, and implementation of abstract data types. In R. Yeh, editor, *Current Trends in Programming Methodology, IV*, chapter 5, pages 80–149. Prentice Hall, New Jersey, 1978.
- [14] G. Grätzer. *Universal algebra*. Springer, New York, 2008. With appendices by Grätzer, B. Jónsson, W. Taylor, R. W. Quackenbush, G. H. Wenzel, and Grätzer and W. A. Lampe. Revised reprint of the 1979 second edition.
- [15] H. Herrlich and G. E. Strecker. *Category theory: an introduction*. Allyn and Bacon Series in Advanced Mathematics. Allyn and Bacon Inc., Boston, Mass., 1973.
- [16] S. Mac Lane. *Categories for the working mathematician*, volume 5. Springer Science & Business Media, 2nd edition, 2013.
- [17] J. Schmidt. A general existence theorem on partial algebras and its special cases. In *Colloq. Math.*, volume 14, pages 73–87, 1966.
- [18] J. Schmidt. A homomorphism theorem for partial algebras. In *Colloq. Math.*, volume 21, pages 5–21, 1970.
- [19] W. Wechler. *Universal algebra for computer scientists*, volume 25. Springer, Berlin, 2012.

PINNs and PINOs for solving PDEs, review and implementation

Adrián Pereira-García^{1,2}, Guillem Rodilla-Alamà^{1,2}, and Wenceslao Tafalla-Domínguez^{1,2}

¹INVEST-MAT Master's Student, Universitat Politècnica de València, Camí de Vera s/n, 46022-València, Spain

²INVEST-MAT Master's Student, Universitat de València, Carrer del Dr. Moliner 50, 46100-Burjassot, Spain

Abstract

This paper presents a comprehensive study on Physics-Informed Neural Networks (PINNs) and Physics-Informed Neural Operators (PINOs) for solving partial differential equations (PDEs). We begin by reviewing the theoretical foundations of PINNs, highlighting their ability to incorporate physical laws into the learning process through physics-informed loss functions. We then introduce PINOs as an extension of PINNs, focusing on their operator-learning capabilities that enable the approximation of solution operators for PDEs. The implementation details of both frameworks are discussed, including network architectures, training strategies, and loss function formulations. We evaluate the performance of PINNs through a benchmark problem involving the heat equation, comparing their accuracy and computational efficiency against traditional numerical methods. Furthermore, we explore the advantages of PINOs in capturing solution operators, state the key differences between PINNs and PINOs, and identify which are the main fields where each methodology can be applied. Our results demonstrate the potential of physics-informed learning approaches in solving PDEs, while also highlighting their limitations and areas for future research.

Introduction and background

Partial differential equations (PDEs) are essential in modeling physical and engineering processes. The heat equation is a key example of PDEs, a parabolic one. It describes how thermal energy spreads in different materials. This equation has applications in many fields, such as thermal engineering, material science, geophysics, and energy systems [20, 21, 5]. The heat equation mathematically formulates the change in temperature distributions within a specific area, following initial and boundary conditions. To simplify our focus, we will examine the classical form of the heat equation in one spatial dimension to evaluate different methods for finding its solution.

Over the past few decades, many classical numerical methods have been developed to estimate the heat equation's solution. Finite difference, finite element, and finite volume methods, along with newer meshless techniques, have proven effective and dependable tools [18, 10]. However, these methods often depend on well-designed meshes or point distributions. They also require problem-specific strategies and iterative solvers, which can become costly with increasing problem complexity, dimensionality, or simulation time [15, 19].

Recently, machine learning techniques have emerged as a promising alternative for approximating solutions to differential equations. In particular, artificial neural networks have shown a remarkable ability to approximate complex nonlinear functions in high-dimensional spaces, as supported by classic approximation theory and thorough theoretical analyses [11, 2]. This has led to their use in scientific computing problems, including solving PDEs, where they have been used as flexible function approximators in data-driven frameworks [22, 13]. However, typical neural network methods are mainly data-driven and need large amounts of training data, which may not always be available or practical. Additionally, since these models do not explicitly apply the physical laws governing the system, they can produce solutions that lack physical consistency [4, 8].

To overcome these challenges, Physics-Informed Neural Networks (PINNs) have been introduced. This innovative framework directly incorporates physical knowledge into the learning process [16]. By embedding the governing equations and initial and boundary conditions into the neural network's loss function, PINNs aim to approximate solutions to partial differential equations while maintaining their physical integrity [1]. This approach offers a flexible, mesh-free alternative to traditional numerical methods, especially for problems with complex geometries, nonlinearities, or limited observational data.

Despite their promising features and results, PINNs also face limitations. To address these issues, Physics-Informed Neural Operators (PINOs) were introduced in 2024 as a natural extension of PINNs [6]. Rather than learning a single solution, PINOs aim to approximate the solution operator that connects input functions to the related solution of the governing PDE. By combining operator learning with physical constraints, PINOs seek to resolve some of the limitations of PINNs while maintaining their physics-informed nature.

The goal of this work is to study PINNs and their operator-based extensions, PINOs. We will implement these methods on a classical heat equation problem and assess whether PINNs and PINOs can deliver competitive or improved approximations. This study aims to clarify the potential benefits and drawbacks of physics-informed learning methods compared to traditional numerical techniques.

Artificial neural networks

This section draws the information of the textbooks by Eiben and Smith [3] and Marsland [9], well-established references in the fields of artificial intelligence and machine

learning.

Artificial neural networks (ANNs) are computational models that take inspiration from the way biological neural systems are structured and function. While they offer a much simplified version of how the human brain works, these networks have shown to be incredibly effective at capturing complex nonlinear relationships.

The basic unit of an ANN lies the artificial neuron. Modeled after biological neurons, this unit takes in a set of input signals, combines them in a linear way using adjustable parameters called weights, adds a bias term, and then applies a nonlinear activation function to generate its output.

In mathematical terms, the output of a single artificial neuron can be represented as

$$y = \sigma \left(\sum_{i=1}^n w_i x_i + b \right)$$

where x_i represents the input variables, w_i are the associated weights, b is the bias term, σ is the activation function, and y is the output of the neuron. The activation function is key to introducing nonlinearity into the model, which is essential for enhancing the network's expressive capabilities.

Common activation functions include the logistic sigmoid, the hyperbolic tangent, and more recently, piecewise linear functions like the rectified linear unit. In scenarios involving function approximation and scientific computing, like ours, smooth activation functions are often favored because they make it easier to compute derivatives and result in smoother approximations.

The weights and biases of the neuron are not fixed a priori; instead, they are learned through a training process. By adjusting these parameters, the neuron can effectively represent a wide range of input-output relationships. While a single neuron has its limitations in terms of representational capacity, it serves as a fundamental building block for more complex networks.

Feedforward neural networks

Practical applications require more complex treatment, that is why they rely on networks formed by combining many neurons into structured architectures. The most common type of artificial neural network is the feedforward neural network, often referred to as the multilayer perceptron [9].

A feedforward neural network consists of an input layer, one or more hidden layers, and an output layer. Each layer contains several neurons, and connections are only permitted from one layer to the next, allowing information to flow in a single direction from input to output. There are no feedback or recurrent connections, which makes both the formulation and the training process simpler.

Let $\mathbf{x} \in \mathbb{R}^n$ denote the input vector. In a matrix terms, the output of the first hidden

layer can be expressed as

$$\mathbf{y}_1 = \sigma(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1)$$

where $\mathbf{W}_1$ is the weight matrix linking the input layer to the first hidden layer, $\mathbf{b}_1$ is the corresponding bias vector, and σ is the activation function used. For the following layers $l = 2, \dots, L$, the network output is calculated recursively as:

$$\mathbf{y}_l = \sigma(\mathbf{W}_l \mathbf{y}_{l-1} + \mathbf{b}_l),$$

This layered structure enables the network to create increasingly abstract representations of the input data as it moves through the layers.

One of the fundamental theoretical results that backs the use of feedforward neural networks is the universal approximation theorem. This theorem asserts that, given some mild conditions on the activation function, a feedforward neural network with at least one hidden layer and a sufficient number of neurons can approximate any continuous function on a compact domain with arbitrary accuracy [2]. This characteristic makes neural networks especially attractive as versatile function approximators.

In the context of scientific computing, especially when tackling the numerical solutions of partial differential equations, feedforward neural networks can be seen as flexible tools that represent unknown solution fields. They offer a versatile, mesh-free way to approximate these solutions. However, the effectiveness of these approximations depends significantly on how the network's parameters are set, which is something we will address in the next section.

Training process and loss functions

In a Feedforward neural network, the key variables are the weights and biases of all the neurons, which are determined during a training process. This is where the network learns to approximate to a target function by minimizing a specific loss function.

When we have a set of training data, represented as $\{(\mathbf{x}^{(j)}, \mathbf{u}^{(j)})\}_{j=1}^N$, with $\mathbf{x}^{(j)}$ being the input variables and $\mathbf{u}^{(j)}$ the corresponding target outputs (or solutions for PDEs), the main goal of training is to find the network parameters, denoted by θ , that reduce the gap between the network's predictions $\mathbf{u}_{\text{NN}}(\mathbf{x}^{(j)}; \theta)$ and the actual outputs. A popular choice for the loss function is the mean squared error (MSE), which is defined as:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{j=1}^N \|\mathbf{u}_{\text{NN}}(\mathbf{x}^{(j)}; \theta) - \mathbf{u}^{(j)}\|^2$$

To minimize this loss function, we usually rely on gradient-based optimization methods. The gradients of the loss function concerning the network parameters are calculated efficiently using the backpropagation algorithm. This method employs the chain rule of calculus to send error information from the output layer back through the earlier layers. This process enables us to iteratively tweak the network parameters, gradually lowering the loss over multiple training epochs.

When it comes to regression problems, especially for function approximation tasks, this training approach has shown to be quite effective, provided we have enough quality data to work with. However, the success of the learned model heavily relies on the availability, accuracy, and distribution of the training data, which is not always a given in practice.

Strengths and limitations of classical neural networks

Classical feedforward neural networks have some really attractive features that explain why they are so popular in machine learning and scientific computing. Their adaptable structures and nonlinear activation functions allow them to capture complex relationships without needing a clear analytical form of the underlying function. Moreover, once they're trained, these networks can efficiently evaluate data at any point in the input space, all without a mesh [14].

That said, when it comes to solving PDEs, traditional neural networks do have some significant drawbacks. Their effectiveness heavily relies on having access to large and representative training datasets, which in physics-related problems often means investing in expensive numerical simulations or experiments [7]. On top of that, standard training methods don't ensure that the solutions adhere to the fundamental physical laws, which can lead to results that break differential equations, boundary conditions, or conservation principles, especially in areas with limited data. Further challenges include sensitivity to hyperparameter settings, slow convergence rates, and restricted extrapolation abilities [12].

These limitations highlight the necessity for learning frameworks that blend the powerful capabilities of neural networks with a direct integration of physical knowledge. Physics-Informed Neural Networks are designed to fill this gap by embedding the governing equations and relevant constraints right into the learning process, as we'll explore in the next subsection.

Physics-Informed Neural Networks

PINNs, or Physics-Informed Neural Networks, were introduced as a unique learning framework that merges the powerful capabilities of neural networks with the direct application of physical laws relevant to the problem at hand. The core concept behind PINNs is to use a standard feedforward neural network to approximate the solution of a differential equation, all while ensuring that the training process aligns with the underlying physics, rather than solely depending on available data. This innovative approach was formalized by [16].

Let us denote the unknown solution of a partial differential equation defined over a domain $\Omega \subset \mathbb{R}^d$ as $u(x)$, which is governed by the equation

$$\mathcal{N}[u](x) = 0, \quad x \in \Omega$$

Here, $\mathcal{N}[\cdot]$ represents a differential operator. Within the PINN framework, the solution $u(x)$ is approximated by a neural network $u_{\text{PINN}}(x; \theta)$, which is defined by its weights

and biases θ . It's important to note that the architecture of the network is similar to that of traditional feedforward neural networks; what sets PINNs apart is the way the learning problem is structured.

By leveraging automatic differentiation, we can differentiate the neural network approximation $u_{\text{PINN}}(x; \theta)$ with respect to its inputs, allowing us to directly evaluate the differential operator $\mathcal{N}[\cdot]$ on the output of the network. This leads to the creation of a physics-based residual

$$r(x; \theta) = \mathcal{N}[u_{\text{PINN}}](x; \theta)$$

which measures how well the neural network's predictions satisfies the governing equation. Consequently, the training process aims to find network parameters that not only fit any available data but also minimize the discrepancies with the physical laws represented by the differential equation. This approach enables PINNs to integrate prior knowledge through governing equations, initial conditions, and boundary conditions, resulting in physics-consistent approximations even in scenarios where data is limited or partially missing.

Physics-informed loss function

The standout characteristic of PINNs is how they build their loss function during training. They take a different approach than ANNs by weaving in physical constraints right into their learning goals. These constraints come from the governing differential equations, along with initial and boundary conditions, and they play a crucial role throughout the training process.

In its broadest sense, the physics-informed loss function can be expressed as

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{data}}(\theta) + \mathcal{L}_{\text{PDE}}(\theta) + \mathcal{L}_{\text{BC}}(\theta) + \mathcal{L}_{\text{IC}}(\theta).$$

Here, $\mathcal{L}_{\text{data}}$ measures how well the network's predictions align with the available observations, while $\mathcal{L}_{\text{PDE}}$, $\mathcal{L}_{\text{BC}}$, and $\mathcal{L}_{\text{IC}}$ impose penalties for any deviations from the governing differential equation, boundary conditions, and initial conditions, respectively.

The physics-based loss term is crafted by assessing the residual of the differential equation at various points sampled within the computational domain, often called collocation points. With the neural network approximation $u_{\text{PINN}}(x; \theta)$, the residual is defined as

$$r(x; \theta) = \mathcal{N}[u_{\text{PINN}}](x; \theta),$$

and the associated loss term usually appears as

$$\mathcal{L}_{\text{PDE}}(\theta) = \frac{1}{N_r} \sum_{i=1}^{N_r} |r(x_i; \theta)|^2,$$

where $\{x_i\}_{i=1}^{N_r}$ represents the set of collocation points in the domain. Notably, these points do not need labeled solution data since the governing equation itself acts as the guiding signal.

The evaluation of the residual and its derivatives depends on automatic differentiation, which allows for precise calculation of the neural network's output derivatives in relation to its inputs, all the way down to machine precision. By minimizing the overall loss function, the network learns to deliver solutions that not only align with the available data but also adhere to the fundamental physical laws, resulting in physically consistent approximations, even when data is limited.

Limitations of Physics-Informed Neural Networks

PINNs provide a flexible framework for incorporating physical laws into neural network training. However, despite their conceptual appeal, several practical limitations have been reported in the literature.

One important issue is the low-frequency, or spectral, bias of neural networks. This means that PINNs may approximate smooth and low-frequency components of the solution more easily than high-frequency or multi-scale structures. As a consequence, their performance can deteriorate in PDEs involving sharp gradients, oscillatory solutions, or multiple spatial and temporal scales.

Another difficulty is related to the propagation of information from the initial and boundary conditions into the interior of the computational domain. In long-time simulations or large domains, the influence of these constraints may become weaker during training, which can reduce the accuracy of the approximation away from the regions where the constraints are imposed.

Scalability is also a relevant concern. As the dimensionality of the problem, the size of the domain, or the complexity of the PDE increases, the optimization problem becomes more expensive and harder to stabilize. In practice, the performance of PINNs depends strongly on the sampling of collocation points, the tuning of the different loss weights, the network architecture, and other problem-dependent choices.

These limitations motivate the use of alternative approaches that move from learning a single PDE solution to learning an operator that maps input functions to solutions. This is the main idea behind neural operators and, in particular, Physics-Informed Neural Operators, which are introduced in the following sections based on [6].

Problem Formulation

We consider two broad classes of partial differential equations: stationary systems and dynamical systems. In both cases, the aim is to approximate the corresponding solution operator, that is, the mapping that assigns to each input function the solution of the PDE.

Stationary Problems

Let $D \subset \mathbb{R}^d$ be a bounded domain with boundary ∂D . We consider stationary PDEs of the form

$$\begin{aligned}\mathcal{P}(a, u)(x) &= 0, & x \in D, \\ u(x) &= g(x), & x \in \partial D,\end{aligned}\tag{1}$$

where $a \in \mathcal{A}$ denotes an input function parameterizing the equation, $u \in \mathcal{U}$ is the unknown solution, and g is a prescribed boundary condition. The operator $\mathcal{P} : \mathcal{A} \times \mathcal{U} \rightarrow \mathcal{F}$ is a partial differential operator, possibly nonlinear. Here, $\mathcal{U}$ and $\mathcal{F}$ are Banach spaces of functions, and $\mathcal{A}$ is a subset of a Banach space.

The solution operator associated with the stationary problem is defined as

$$\mathcal{G}^\dagger : \mathcal{A} \rightarrow \mathcal{U}, \quad \mathcal{G}^\dagger(a) = u.\tag{2}$$

Dynamical Problems

We next consider time-dependent PDEs defined on the space-time domain $D \times (0, T]$. The dynamical system is given by

$$\begin{aligned}\frac{\partial u}{\partial t}(x, t) &= \mathcal{R}(u)(x, t), & (x, t) \in D \times (0, T], \\ u(x, t) &= g(x, t), & (x, t) \in \partial D \times (0, T], \\ u(x, 0) &= a(x), & x \in D.\end{aligned}\tag{3}$$

Here, the input function $a \in \mathcal{A}$ corresponds to the initial condition of the system, $u(\cdot, t) \in \mathcal{U}$ is the unknown solution for $t > 0$, and g is a prescribed boundary condition. The operator $\mathcal{R}$ denotes the differential operator governing the dynamics. Typical examples include the Burgers equation and the incompressible Navier–Stokes equations.

The solution operator associated with the dynamical problem is defined as

$$\mathcal{G}^\dagger : \mathcal{A} \rightarrow \mathcal{C}([0, T]; \mathcal{U}), \quad \mathcal{G}^\dagger(a) = u.\tag{4}$$

In both the stationary and dynamical settings, the objective is to approximate the solution operator $\mathcal{G}^\dagger$ rather than individual solution realizations. This operator-learning perspective forms the foundation of neural operators and, in particular, Physics-Informed Neural Operators.

Neural Operators

Neural operators aim to approximate the solution operators defined in (2) and (4). Unlike standard neural networks, which approximate functions between finite-dimensional spaces, neural operators are designed to approximate mappings between function spaces.

A neural operator can be written in the form

$$\mathcal{G}_\theta = \mathcal{Q} \circ \sigma(W_{L-1} + \mathcal{K}_{L-1}) \circ \cdots \circ \sigma(W_0 + \mathcal{K}_0) \circ \mathcal{P}_{\text{lift}},\tag{5}$$

where $\mathcal{P}_{\text{lift}}$ is a lifting operator that maps the input function into a higher-dimensional feature space, and $\mathcal{Q}$ is a projection operator that maps the final feature representation to the target solution space.

Equivalently, the action of the neural operator can be described recursively as

$$v_0(x) = \mathcal{P}_{\text{lift}}(a)(x), \quad (6)$$

$$v_{\ell+1}(x) = \sigma(W_\ell v_\ell(x) + (\mathcal{K}_\ell v_\ell)(x) + b_\ell(x)), \quad \ell = 0, \dots, L-1, \quad (7)$$

and

$$\mathcal{G}_\theta(a)(x) = \mathcal{Q}(v_L)(x). \quad (8)$$

Here, W_ℓ are local linear transformations, $\mathcal{K}_\ell$ are nonlocal kernel operators, b_ℓ are bias terms, and σ is a fixed nonlinear activation function. The trainable parameters θ are those defining the lifting operator, the projection operator, the local linear maps, the kernel operators, and the biases.

Kernel Integral Operators

A common choice for the nonlocal operator $\mathcal{K}_\ell$ is a kernel integral operator. Let

$$\kappa^{(\ell)} \in C(D \times D, \mathbb{R}^{d_{\ell+1} \times d_\ell})$$

and let ν be a Borel measure on D . The kernel integral operator is defined by

$$(\mathcal{K}_\ell v_\ell)(x) = \int_D \kappa^{(\ell)}(x, y) v_\ell(y) d\nu(y), \quad x \in D. \quad (9)$$

In the Fourier Neural Operator setting, this kernel operator is parameterized in Fourier space and implemented through a finite number of Fourier modes. In this case, one can write

$$(\mathcal{K}_\ell v_\ell)(x) = \text{FFT}^{-1}(R_\ell \cdot \text{FFT}(v_\ell))(x), \quad (10)$$

where FFT and FFT^{-1} denote the discrete Fourier transform and its inverse, and R_ℓ contains learnable weights acting on selected Fourier modes.

Training neural operators

Our goal is to train a neural operator $\mathcal{G}_\theta$ that approximates the solution operator $\mathcal{G}^\dagger$. This is achieved by solving an optimization problem in which the parameters θ are obtained by minimizing a loss functional that combines data fidelity with physical consistency.

Data loss. We assume access to a dataset $\{a_j, u_j\}_{j=1}^N$, where $u_j = \mathcal{G}^\dagger(a_j)$ and the inputs a_j are independent samples drawn from a probability measure μ supported on $\mathcal{A}$. For the stationary problem, the pointwise data loss is defined as

$$\mathcal{L}_{\text{data}}(u, \mathcal{G}_\theta(a)) := \|u - \mathcal{G}_\theta(a)\|_{\mathcal{U}}^2 = \int_D |u(x) - \mathcal{G}_\theta(a)(x)|^2 dx. \quad (11)$$

The corresponding operator-level data loss is obtained by averaging over the input distribution,

$$\mathcal{J}_{\text{data}}(\mathcal{G}_\theta) := \mathbb{E}_{a \sim \mu} [\mathcal{L}_{\text{data}}(u, \mathcal{G}_\theta(a))] \approx \frac{1}{N} \sum_{j=1}^N \int_D |u_j(x) - \mathcal{G}_\theta(a_j)(x)|^2 dx. \quad (12)$$

For time-dependent problems, the same idea is applied over the corresponding space-time domain.

Physics-informed loss: stationary problems. To enforce consistency with the underlying PDE, PINOs incorporate the governing equation directly into the loss function. For the stationary problem (1), the residual is obtained by evaluating the operator $\mathcal{P}$ at the neural operator prediction $\mathcal{G}_\theta(a)$. The physics-informed loss is then defined as

$$\mathcal{L}_{\text{phys}}(a, \mathcal{G}_\theta(a)) := \int_D |\mathcal{P}(a, \mathcal{G}_\theta(a))(x)|^2 dx + \alpha \int_{\partial D} |\mathcal{G}_\theta(a)(x) - g(x)|^2 dS, \quad (13)$$

where $\alpha > 0$ balances the relative importance of the boundary condition and the interior PDE residual.

By averaging over the input distribution, we obtain the operator-level physics loss

$$\mathcal{J}_{\text{phys}}(\mathcal{G}_\theta) := \mathbb{E}_{a \sim \mu} [\mathcal{L}_{\text{phys}}(a, \mathcal{G}_\theta(a))]. \quad (14)$$

Physics-informed loss: dynamical problems. The same construction applies to time-dependent PDEs of the form (3). In this case, the physical residual is

$$\frac{\partial \mathcal{G}_\theta(a)}{\partial t}(x, t) - \mathcal{R}(\mathcal{G}_\theta(a))(x, t). \quad (15)$$

The corresponding physics-informed loss is defined over the space-time domain as

$$\begin{aligned} \mathcal{L}_{\text{phys}}(a, \mathcal{G}_\theta(a)) &:= \int_0^T \int_D \left| \frac{\partial \mathcal{G}_\theta(a)}{\partial t}(x, t) - \mathcal{R}(\mathcal{G}_\theta(a))(x, t) \right|^2 dx dt \\ &+ \alpha \int_0^T \int_{\partial D} |\mathcal{G}_\theta(a)(x, t) - g(x, t)|^2 dS dt \\ &+ \beta \int_D |\mathcal{G}_\theta(a)(x, 0) - a(x)|^2 dx. \end{aligned} \quad (16)$$

Here, $\alpha, \beta > 0$ are weighting parameters that balance the boundary and initial condition terms with respect to the PDE residual.

The operator loss $\mathcal{J}_{\text{phys}}(\mathcal{G}_\theta)$ is again obtained by taking the expectation with respect to the input distribution μ .

Total training objective. The final loss functional used to train the Physics-Informed Neural Operator is given by

$$\mathcal{J}_{\text{total}}(\mathcal{G}_\theta) := \mathcal{J}_{\text{data}}(\mathcal{G}_\theta) + \lambda \mathcal{J}_{\text{phys}}(\mathcal{G}_\theta), \quad (17)$$

where $\lambda > 0$ balances data fitting and physical consistency.

Derivatives of Neural Operators

For dynamical problems, the physics-informed loss requires the evaluation of the time derivative

$$\partial_t \mathcal{G}_\theta(a)(x, t).$$

The treatment of this derivative depends on the formulation of the neural operator.

In classical PINNs, the solution is represented directly as a neural network $u_\theta(x, t)$, and derivatives can be computed by automatic differentiation with respect to the input variables. In neural operators, the output is a function or a time-dependent solution field. Therefore, in discrete-time formulations, temporal derivatives are not directly available and must be approximated numerically. In continuous-time formulations, time can instead be included as an input variable, allowing the time derivative to be computed by automatic differentiation.

Discrete-time formulation. In the discrete-time or autoregressive formulation, the neural operator is trained to map the solution at one time step to the solution at the next one. The time derivative is then approximated using finite differences between consecutive outputs. This strategy avoids explicit differentiation with respect to time, but it can lead to error accumulation when the model is rolled out over long time intervals.

Continuous-time formulation. Alternatively, the neural operator can include time as an explicit input variable, yielding a continuous-in-time representation of the solution. In this setting, the time derivative can be evaluated through automatic differentiation, and the PDE residual can be enforced directly. However, this approach increases the computational cost and may be harder to scale to long-time dynamics.

Both formulations involve a trade-off between numerical stability, computational efficiency, and accuracy. The most appropriate choice depends on the structure of the PDE, the available data, and the temporal resolution required.

From Theory to Computation

We have introduced the operator-learning framework underlying neural operators and their physics-informed extensions. We formulated stationary and dynamical PDEs from an operator perspective and described how data-driven and physics-informed losses can be combined to train Physics-Informed Neural Operators.

In practice, the performance of these methods depends strongly on implementation choices, including the discretization of the domain, the treatment of derivatives, the sampling strategy, and the optimization procedure. In the following section, we illustrate these computational aspects through a PINN implementation for the one-dimensional heat equation, and then discuss how the workflow changes when moving from PINNs to PINOs.

How can we solve PDEs using PINNs and PINOs?

PINNs

Why PINNs?

PINNs are being used in recent years as a reliable tool for solving PDEs by incorporating the governing physical laws directly into the training process [16, 1, 4]. This physics-based regularization reduces the dependence on labeled datasets compared to purely data-driven approaches, and makes PINNs applicable to nonlinear, high-dimensional, and multiphysics problems within a unified formulation. The mesh-free character of the method is particularly convenient for problems involving complex geometries and for settings where constructing a suitable grid is nontrivial. However, these advantages come with trade-offs: PINNs do not always outperform classical solvers, and their competitiveness depends on the problem at hand, as it is discussed below.

Are PINNs better than traditional numerical methods?

A direct comparison between PINNs and classical numerical solvers is not straightforward and must be interpreted carefully. Based on the literature [17, 4, 12], some qualitative observations can be made. Traditional methods are better in accuracy and interpretability for problems where efficient, well-tested discretizations exist. Data-driven approaches require large datasets but generalize well when data are abundant. PINNs cover the gap between these two approaches: they do not require labeled data in the interior of the domain (the PDE itself acts as the training signal), they yield mesh-free and differentiable approximations, and they handle inverse problems and incomplete data settings naturally. On the other hand, PINNs are generally slower to train than classical solvers on standard benchmarks, and their accuracy may not match that of high-order finite element or finite difference schemes in well-posed, regular settings. For these reasons, PINNs are best regarded as a complementary tool rather than a universal replacement for classical methods, and are especially well-suited for data-scarce, high-dimensional, or complex-geometry problems where traditional approaches face significant challenges.

PINNs workflow

For solving a PDE using PINNs, the following workflow is typically followed:

- **Problem Definition:** Clearly define the PDE, including the domain, boundary conditions, and any initial conditions.
- **Neural Network Architecture:** Design a neural network architecture suitable for the problem at hand. This involves selecting the number of layers, neurons per layer, and activation functions.
- **Loss Function Formulation:** Construct a loss function incorporating the residuals of the PDE, boundary conditions, and initial conditions. This loss quantifies

how well the neural network satisfies the governing equations and constraints.

- **Model Training:** Train the neural network using optimization algorithms (e.g., Adam, L-BFGS) to minimize the loss function by iteratively updating the network weights based on computed gradients.
- **Model Evaluation and Validation:** After training, evaluate the PINN by comparing its predictions against known solutions or benchmark problems, computing error metrics and visualizing the results.

Implementation Example

In order to illustrate the PINNs workflow described above, we present a simple implementation example for solving the one-dimensional heat equation using PINNs.

Problem Definition

We consider the one-dimensional heat equation on the domain $x \in [0, 1], t \in [0, 1]$, with thermal diffusivity $\alpha = 0.01$:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \quad x \in [0, 1], \quad t \in [0, 1], \quad (18)$$

subject to the boundary and initial conditions

$$u(0, t) = u(1, t) = 0, \quad u(x, 0) = \sin(\pi x). \quad (19)$$

The exact analytical solution is

$$u(x, t) = e^{-\alpha\pi^2 t} \sin(\pi x), \quad x \in [0, 1], \quad (20)$$

which serves as ground truth for validation.

Neural Network Architecture

A fully connected feedforward neural network takes (x, t) as inputs and outputs the scalar temperature $u(x, t)$. The architecture consists of three hidden layers with 50 neurons each, representing a reasonable compromise between expressiveness and computational cost for this low-dimensional parabolic benchmark, consistent with configurations commonly adopted in the literature. The hyperbolic tangent ($\tanh$) is used as the activation function, owing to its smoothness and continuous differentiability—properties essential for the accurate computation of spatial and temporal derivatives through automatic differentiation.

Loss Function Formulation

Since no observational data are used, the loss relies entirely on physical constraints:

$$\mathcal{L} = \mathcal{L}_{\text{PDE}} + \mathcal{L}_{\text{BC}} + \mathcal{L}_{\text{IC}}, \quad (21)$$

with $N_f = 10,000$ collocation points for the PDE residual, $N_b = 200$ boundary points, and $N_0 = 200$ initial condition points, all sampled uniformly. The three terms are:

$$\mathcal{L}_{\text{PDE}} = \frac{1}{N_f} \sum_{i=1}^{N_f} \left| \frac{\partial u}{\partial t}(x_i, t_i; \theta) - \alpha \frac{\partial^2 u}{\partial x^2}(x_i, t_i; \theta) \right|^2, \quad (22)$$

$$\mathcal{L}_{\text{BC}} = \frac{1}{N_b} \sum_{i=1}^{N_b} \left(|u(0, t_i; \theta)|^2 + |u(1, t_i; \theta)|^2 \right), \quad (23)$$

$$\mathcal{L}_{\text{IC}} = \frac{1}{N_0} \sum_{i=1}^{N_0} |u(x_i, 0; \theta) - \sin(\pi x_i)|^2. \quad (24)$$

Model Training

The network is trained using the Adam optimizer with an initial learning rate of 0.001 and a decay factor of 0.9 applied every 1000 epochs, for a total of 6000 epochs. No early stopping criterion is used beyond the maximum epoch count, which is appropriate given the simplicity of this benchmark. Gradients are computed via automatic differentiation and backpropagation at each epoch.

Model Evaluation and Validation

Figure 1 shows the evolution of the training loss over 6000 epochs. The loss decreases from approximately 10^{-1} to below 10^{-4} , with periodic spikes that coincide with the learning rate decay steps—this is expected behavior of the scheduler and does not indicate training instability. The plateau toward the end of training suggests that the model has reached convergence within the allocated budget.

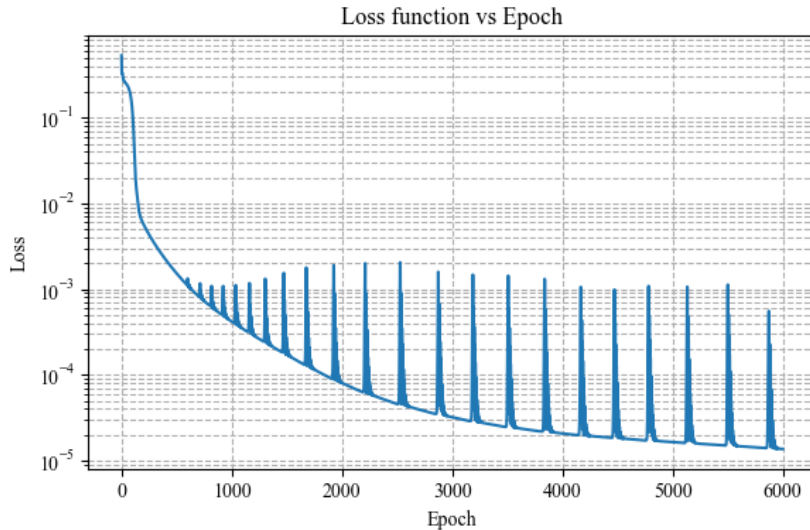


Figure 1: Loss evolution during training.

After training, the PINN solution and the analytical solution are evaluated on a uniform grid over $[0, 1] \times [0, 1]$. Figure 2 displays both solutions: the PINN prediction

is shown on the left and the analytical solution on the right. Both heatmaps correctly capture the sinusoidal initial profile and its exponential decay over time.

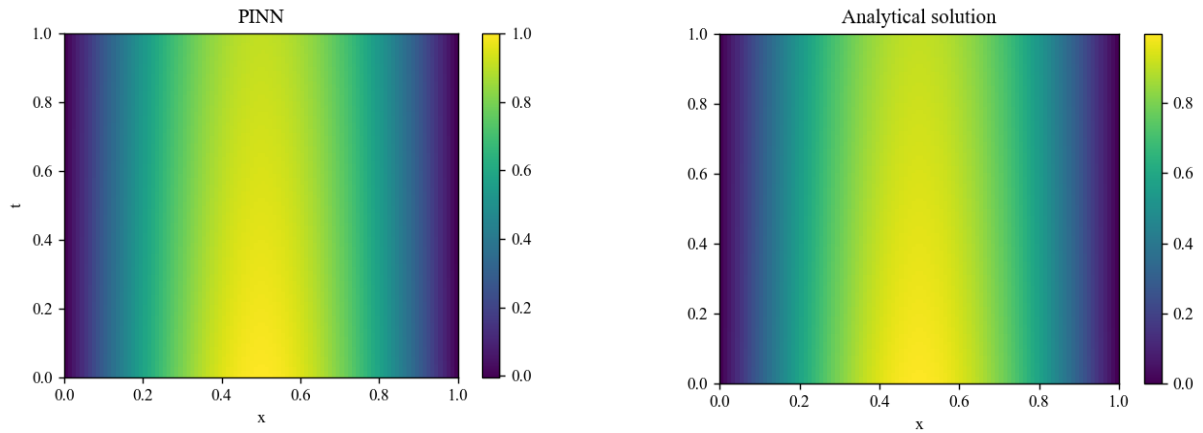


Figure 2: Solutions obtained by the PINN (left) and analytically (right).

Figure 3 shows the pointwise absolute error $|u_{\text{PINN}} - u_{\text{analytical}}|$. The maximum error is approximately $4 \cdot 10^{-3}$, consistent with the colorbar scale of the figure. The error is largest near the center of the spatial domain at intermediate times, which is consistent with the known spectral bias of PINNs toward low-frequency solution components.

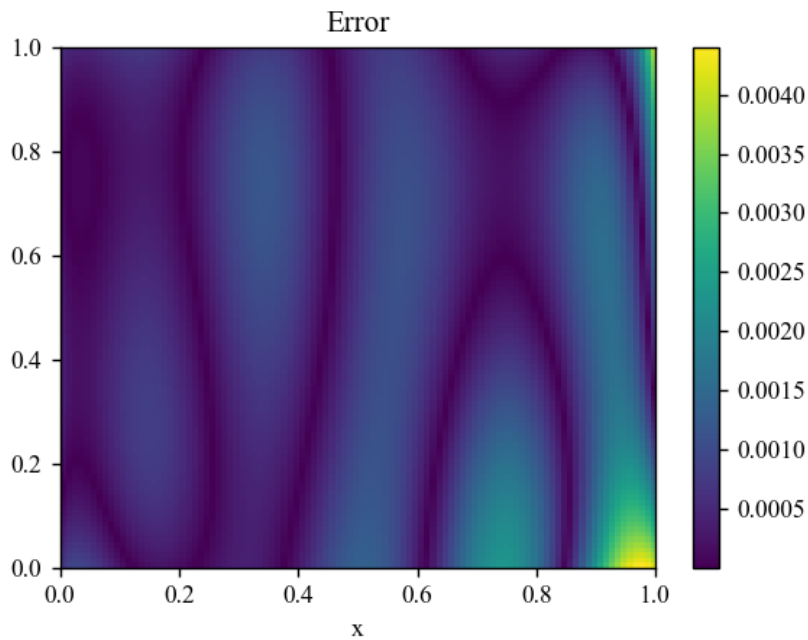


Figure 3: Pointwise absolute error between the PINN and the analytical solution.

Figure 4 compares both solutions at five time snapshots ($t = 0, 0.25, 0.5, 0.75, 1.0$). At $t = 0$ the PINN closely matches the prescribed initial condition $\sin(\pi x)$; for later times, the predicted profiles agree well with the analytical decay curves, with small but visible deviations at the peak.

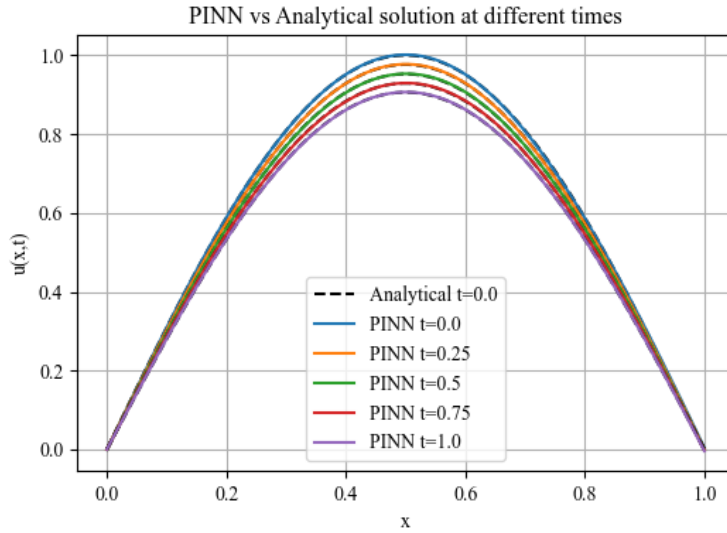


Figure 4: Comparison of PINN and analytical solutions at different times.

For a global accuracy metric we compute the root mean squared error (RMSE) between the PINN-predicted solution and the analytical solution over a dense evaluation grid:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N \left(u_{\text{PINN}}(x_i, t_i) - u_{\text{analytical}}(x_i, t_i) \right)^2}, \quad (25)$$

where N is the total number of evaluation points. For this implementation the RMSE is approximately 8.71×10^{-4} , confirming a high level of accuracy. These results show that, for a simple one-dimensional heat equation, the PINN is able to recover the analytical solution with low error. Whether this accuracy extends to more complex settings requires further experimentation and is left for future work. The gap between PINNs and classical solvers could also be narrowed by supplementing the physics-based loss with observational data, giving rise to so-called data-enhanced PINNs.

Transition to PINOs

Why PINOs?

While PINNs are effective for solving individual PDE instances, they must be fully retrained whenever the problem parameters change initial conditions, boundary conditions, or source terms. This is a significant practical limitation in parametric or multi-query settings. PINOs emerge as a solution for this since they learn a solution operator rather than a single solution, enabling generalization across a family of problems without retraining.

We can list some of the main advantages of PINOs over PINNs. First, they offer *parameter space generalization*: a trained PINO handles a wide range of parameterized PDEs without retraining, which is particularly efficient when multiple simulations across different parameter settings are required. Second, they adapt more naturally to complex geometries than PINNs. Third, once trained, PINOs support *real-time inference*, which is relevant, for example, for control systems. Finally, they efficiently manage *multi-query scenarios*, handling multiple input conditions or source terms within a single trained model.

Differences between PINNs and PINOs

The fundamental distinction lies in the learning target. A PINN approximates a single solution $u(x, t)$ for a specific PDE instance, while a PINO learns the mapping $\mathcal{G}^\dagger: a \mapsto u$ from input functions (initial conditions, boundary conditions, source terms) to solutions. This allows PINOs to generalize across instances of the same PDE with varying inputs, at the cost of a more involved training procedure requiring a dataset of input-output function pairs.

This difference does not make one method superior to the other in every situation. If only one or a few PDE instances need to be solved and data are scarce, a PINN could be a better solution. If many instances must be solved across a parameter space and training data are available, a PINO is more efficient. Both methods have appropriate application fields and complementary strengths.

The table below summarizes approximate speedups of neural methods over classical solvers across several domains, based on results reported in the literature [23]. These are order-of-magnitude estimates that depend on the specific method, architecture, problem size, and hardware, and should be interpreted as qualitative indicators rather than guaranteed performance gains.

Table 1: Approximate speedup of neural methods over classical solvers across domains. Order-of-magnitude estimates from the literature [23].

Domain	Representative methods	Reported speedup
Fluid Dynamics	FNO, DeepONet, PINO	$10^3\text{--}10^5\times$
Solid Mechanics	PINN, DeepONet, GNO	$10^2\text{--}10^3\times$
Heat Transfer	PI-DeepONet, PINN	$10^3\text{--}10^4\times$
Electromagnetics	PINN, MaxwellNet	$10^2\text{--}10^3\times$

PINOs workflow

The workflow for implementing a PINO broadly follows that of PINNs but with key differences in architecture and training data requirements:

- **Problem Definition:** Specify the PDE and the range of input functions (initial conditions, boundary conditions, source terms) that the PINO must learn to map to solutions.

- **Operator Network Architecture:** Select an architecture capable of learning the operator mapping, typically a DeepONet or a Fourier Neural Operator (FNO).
- **Loss Function Formulation:** Construct a loss that measures discrepancy between predicted and reference solutions over a diverse set of training input functions, and optionally adds a physics-informed residual term. Note that including the PDE residual is what distinguishes a true PINO from a purely supervised neural operator.
- **Model Training:** Minimize the loss using standard optimization algorithms by iteratively updating network weights based on computed gradients.
- **Model Evaluation and Validation:** Assess the trained PINO on unseen input functions and compare predictions against known or benchmark solutions.

Conclusions

This work has reviewed the theoretical foundations and practical implementation of PINNs and PINOs for solving PDEs. The main idea of PINNs is to approximate a single PDE solution by minimizing a loss function that penalizes data mismatch and physical residuals at the same time, using automatic differentiation to evaluate differential operators on the network output. In the case of PINOs, the learning target shifts from individual solutions to the solution operator itself, enabling generalization across multiple parameters.

The benchmark experiment on the one-dimensional heat equation with initial condition $\sin(\pi x)$ illustrates the PINN workflow concretely. The trained model achieves an RMSE of approximately 8.71×10^{-4} with respect to the analytical solution $u(x, t) = e^{-\alpha\pi^2 t} \sin(\pi x)$, demonstrating that PINNs can accurately recover simple analytical solutions for this kind of problem. More complex or higher-dimensional settings would require additional experiments and are left for future work.

Overall, both PINNs and PINOs provide useful alternatives for combining machine learning techniques with traditional PDE-solving approaches. In practice, these methods become especially interesting in problems involving incomplete data, inverse formulations, or geometries that are difficult to discretize. Improving the robustness and accuracy of these methods compared with established numerical solvers remains an open research challenge.

References

- [1] Hubert Baty. A hands-on introduction to physics-informed neural networks for solving partial differential equations with benchmark tests taken from astrophysics and plasma physics. *arXiv preprint*, 2024.

- [2] Ronald DeVore, Guergana Petrova, and Przemyslaw Wojtaszczyk. Neural networks and approximation theory. *Proceedings of the National Academy of Sciences*, 118(45):e2014637118, 2021.
- [3] Agoston E Eiben and James E Smith. *Introduction to evolutionary computing*. Springer, 2015.
- [4] A. Farea, O. Yli-Harja, and F. Emmert-Streib. Understanding physics-informed neural networks: Techniques, applications, trends, and challenges. *AI*, 5(3):1534–1557, 2024.
- [5] R. S. V. Kumar, I. E. Sarris, G. Sowmya, and A. Abdulrahman. Iterative solutions for the nonlinear heat transfer equation of a convective-radiative annular fin with power law temperature-dependent thermal properties. *Symmetry*, 15(6):1204, 2023.
- [6] Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *ACM / IMS J. Data Sci.*, 1(3), May 2024.
- [7] Kevin Linka and Ellen Kuhl. A new family of constitutive artificial neural networks towards automated model discovery. *Computer Methods in Applied Mechanics and Engineering*, 403:115731, 2023.
- [8] Ziti Liu, Yang Liu, Xunshi Yan, Wen Liu, Han Nie, Shuaiqi Guo, and Chen-an Zhang. Automatic network structure discovery of physics informed neural networks via knowledge distillation. *Nature Communications*, 16(1):9558, 2025.
- [9] Stephen Marsland. *Machine learning: an algorithmic perspective*. Chapman and Hall/CRC, 2011.
- [10] Davide Miotti, Roberto Zamolo, and Enrico Nobile. A fully meshless approach to the numerical simulation of heat conduction problems over arbitrary 3d geometries. *Energies*, 14(5):1351, 2021.
- [11] Thomas Most. *Approximation of Complex Nonlinear Functions by Means of Neural Networks*. VDI Verlag, 2005.
- [12] Karthika Nasir, Aradhana Reva, and Jai Sekhar. Embedding physics into deep learning: A structured review of physics-informed neural networks. *Preprints*, April 2025.
- [13] Samuel M. Nuugulu, Kailash C. Patidar, and Divine T. Tarla. Data driven neural network approaches for pricing options. *Physica D: Nonlinear Phenomena*, 484:134992, 2025.
- [14] Varun Kumar Ojha, Ajith Abraham, and Václav Snášel. Metaheuristic design of feedforward neural networks: A review of two decades of research. *Engineering Applications of Artificial Intelligence*, 60:97–116, April 2017.

- [15] Vivek G. Patel and Nikunj V. Rachchh. Meshless method – review on recent developments. *Materials Today: Proceedings*, 26:1598–1603, 2020.
- [16] Maziar Raissi, Paris Perdikaris, and George E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
- [17] Zhiyuan Ren, Shijie Zhou, Dong Liu, and Qihe Liu. Physics-informed neural networks: A review of methodological evolution, theoretical foundations, and interdisciplinary frontiers toward next-generation scientific computing. *Applied Sciences*, 15(14), 2025.
- [18] Rajesh Sharma, Vijay Jadon, and B. Singh. A review on the finite element methods for heat conduction in functionally graded materials. *Journal of The Institution of Engineers (India): Series C*, 96:73–81, 2014.
- [19] Eitan Tadmor. Mathematics of partial differential equations: From classical to modern methods. *Bulletin of the American Mathematical Society*, 49(4):507–554, 2012.
- [20] H. Wang, X. Chen, E. Koenders, Y. Dai, X. Huang, Q. Ai, and Y. Yuan. Transient nonlinear heat conduction in concrete structures: A semi-analytical approach. *Entropy*, 25(4):583, 2023.
- [21] Jinhong Yu, Chen An, Yangye He, and Jian Su. Transient nonlinear heat conduction in a slab with temperature-dependent thermal conductivity: Integral transform and lumped model solutions. *International Communications in Heat and Mass Transfer*, 158:107886, 2024.
- [22] Wenshu Zha, Daolun Li, Luhang Shen, Wen Zhang, and Xuliang Liu. Review of neural network-based methods for solving partial differential equations. *Chinese Journal of Theoretical and Applied Mechanics*, 54(3):543–556, 2022.
- [23] Zhuo Zhang, Xiong Xiong, Sen Zhang, Yuan Zhao, and Xi Yang. Physics-informed neural networks and neural operators for parametric pdes: A human-ai collaborative analysis, 2025.

Qualitative study of a tumor growth model

Cristina Vilar-Medina*, M. Carmen Martí, and Antonio Baeza

Departament de Matemàtiques, Universitat de València, Av. Vicent Andrés Estellés 19, 46100–Burjassot, Spain

Abstract

Tumor progression depends not only on the intrinsic characteristics of the cancerous cells but also on the bidirectional complex interactions within the tumor microenvironment. Among these interactions, lymphocytes or immune cells, play an important role.

In this work, we study a simplified version of the mathematical model developed by Sung et al. in *A tumor-immune interaction model for hepatocellular carcinoma based on measured lymphocyte counts in patients undergoing radiotherapy*, Radiotherapy and Oncology 151. This model describes the stimulating effect of malignant cells on the immune system, as well as the impact of lymphocytes on the tumor. We describe and study the dynamics of the model, detailing their equilibrium points and their stability. Some numerical simulations will be presented to validate the theoretical results and illustrate the behavior of the system.

Introduction

In the past years, biomedicine research has achieved a lot of advances, improving the quality of life of many people. Still, the incidence of some diseases is increasing globally. One of them is cancer, a disease caused by a group of cells with a proliferation and death disorder, which invade the contiguous tissues and can end up in metastasis.

According to the Global Cancer Observatory of the World Health Organization, in 2022, almost 20 million people were diagnosed with cancer [2]. The high incidence of this disease, coupled with its biological complexity and socioeconomic burden, makes necessary its study from different multidisciplinary approaches.

It is the aim of this work to study the biologic mechanisms that intervene in the evolution of cancer from a mathematical point of view. To do so, a submodel of the model presented in [5] is considered. In [5], the authors proposed a model given by a system of ordinary differential equations (ODEs) for the evolution of a hepatocellular

*cristina.vilar@uv.es

carcinoma under radiotherapy. In their model two types of tumor cells, from the primary tumor and from a metastatic one, the irradiated cells and the lymphocytes were considered.

The simplified submodel considered in this work studies the interaction only between the primary tumor cells and the immune cells, without any treatment. This submodel describes the initial stage of the disease, where the body identifies the tumor cells as malignant and starts producing lymphocytes to attack them.

A qualitative analysis of the submodel will be presented, detailing their equilibrium points and studying their stability. This will translate into some necessary conditions involving the parameters of the model for the disease to be stabilized without further treatment.

This work is organized as follows: In Section 1 we present the model, explaining in detail the biological meaning of each equation. The qualitative study of the model presented will be performed in Section 2. Later, in Section 3 some simulations to exemplify the results in Section 2 will be shown. Finally, some conclusions will be drawn.

Mathematical Model

In this section we derive in detail the equations of the model describing the interaction between tumorous and immune cells.

The number of tumor cells, denoted by T , is commonly modeled using the Malthus law [1]. This equation describes the exponential nature of their growth, with rate a , when no process diminishes their growth.

The body generates immune system cells called lymphocytes (L) in response to the presence of malignant cells [4]. Their creation follows a sigmoidal reaction against tumor cells. Thus, this reaction is supposed to follow a Monod form [3], a mathematical model initially used to describe the growth of microorganisms. This term is given by

$$\frac{T}{g + T}L,$$

where g is a semi-saturation constant regarding T , i.e., g represents the value at which the population of tumor cells reaches half of its maximum capacity. Since not every lymphocyte will be effective battling the tumor cells, a constant ω_1 is included representing the immune system effectiveness towards the tumor.

Hence, the equation that describes the growth of the number tumor cells is given by:

$$\frac{dT}{dt} = aT - \omega_1 \frac{T}{g + T}L. \quad (1)$$

As we have previously said, the presence of tumor cells encourages lymphocytes production. We suppose that this process follows a Monod form, multiplied by a constant ω_2 that represents the immune system response to the cancer. Lymphocytes are

cells from the immune system and they target different tasks, such as regenerating the bone marrow or helping against diseases. Thereby, the body produces these cells whether there is a tumor or not. We denote the production rate of lymphocytes as s and their decay rate due to natural processes as f .

Summing up, the equation modelling the evolution of the population of lymphocytes is given by:

$$\frac{dL}{dt} = \omega_2 \frac{T}{g+T} L + s - fL. \quad (2)$$

Finally, the submodel considered in this work is given by equations (1) and (2):

$$\begin{cases} \frac{dT}{dt} = aT - \omega_1 \frac{T}{g+T} L, \\ \frac{dL}{dt} = \omega_2 \frac{T}{g+T} L + s - fL. \end{cases} \quad (3)$$

We assume that the constants $a, \omega_1, \omega_2, g, s$ and f are strictly positive and not zero, according with their biological meaning.

Qualitative Study

We start the qualitative study of the model obtaining the equilibrium points of the system (3) solving the following system of equations:

$$\begin{cases} 0 = aT - \omega_1 \frac{T}{g+T} L, \\ 0 = \omega_2 \frac{T}{g+T} L + s - fL. \end{cases} \quad (4)$$

From the first equation of (4), we obtain different possibilities for T and L :

1. If $T \neq 0$, then:

$$L = \frac{a(g+T)}{\omega_1}. \quad (5)$$

Substituting this value of L in the second equation of (4), we get:

$$0 = \left(\omega_2 \frac{T}{g+T} - f \right) \frac{a(g+T)}{\omega_1} + s = \frac{\omega_2 a T}{\omega_1} - \frac{a f (g+T)}{\omega_1} + s. \quad (6)$$

(a) On the one hand, if $f \neq \omega_2$, the expressions of T and L are given by:

$$\begin{aligned} T &= \frac{s\omega_1 - afg}{a(f - \omega_2)}, \\ L &= \frac{s\omega_1 - ag\omega_2}{f\omega_1 - \omega_1\omega_2}. \end{aligned}$$

Therefore, one equilibrium point of the model (3) is:

$$P_1 = (T^*, L^*) = \left(\frac{s\omega_1 - afg}{a(f - \omega_2)}, \frac{s\omega_1 - ag\omega_2}{f\omega_1 - \omega_1\omega_2} \right). \quad (7)$$

(b) On the other hand, when $f = \omega_2$, the expression (6) holds as long as $f = \frac{s\omega_1}{ga}$.

Hence, the result is a line of fixed points depending on the value of T :

$$L_2 = (T^*, L^*) = \left(T^*, \frac{a(g + T^*)}{\omega_1} \right). \quad (8)$$

Both P_1 and L_2 represent scenarios where the tumor does not disappear, their cells stabilize without treatment, only with the effect of the immune system. It is important to note that the equilibrium points in L_2 are unlikely scenarios in real life, since $f = \omega_2$, but they are theoretical results of the model.

2. If $T = 0$, the respective equilibrium point is:

$$P_3 = (T^*, L^*) = \left(0, \frac{s}{f} \right). \quad (9)$$

This case represents the ideal outcome for the patient, where the tumor completely disappears and lymphocytes are only created to meet the body requirements, so they stabilize at s/f .

To study the stability of these equilibrium points, we need the expression of the eigenvalues of the Jacobian matrix of the system (3) evaluated on each of them.

The Jacobian matrix of the system (3) is given by

$$J(T, L) = \begin{bmatrix} a - \frac{\omega_1 g L}{(g+T)^2} & -\frac{\omega_1 T}{g+T} \\ \frac{\omega_2 g L}{(g+T)^2} & \frac{\omega_2 T}{g+T} - f \end{bmatrix}. \quad (10)$$

Solving its characteristic equation, $|J - \lambda I| = 0$, we obtain that the eigenvalues λ_1 and λ_2 of J are the solutions of the second degree equation:

$$0 = \lambda^2(g + T)^2 + \lambda((f - a)(g + T)^2 - T\omega_2(g + T) + gL\omega_1) - af(g + T)^2 + aT\omega_2(g + T) + fgL\omega_1. \quad (11)$$

1. **Case 1.** First, we take the fixed point P_1 in (7) and substitute its expression in (11). After some calculations we have that:

$$\lambda_{1,2} = -R \pm \frac{\sqrt{\Delta}}{2(s\omega_1 - ag\omega_2)}, \quad (12)$$

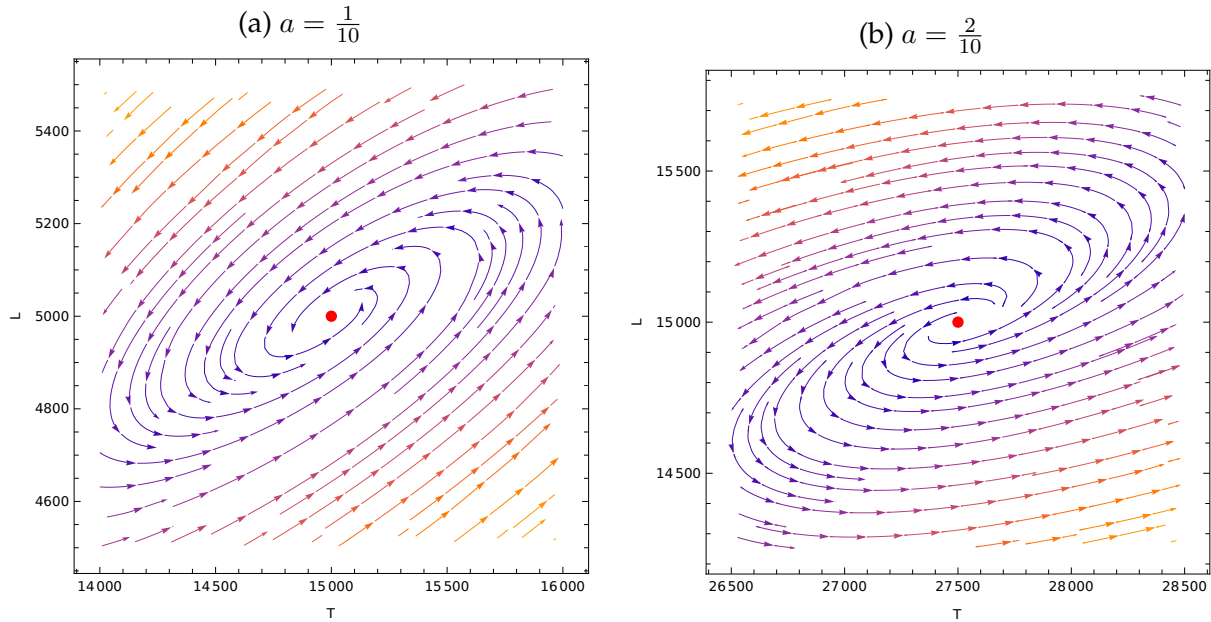
with

$$\Delta = (a^2fg + s\omega_1(f - a - \omega_2))^2 + 4a(f - \omega_2)(afg - s\omega_1)(ag\omega_2 - s\omega_1),$$

$$R = \frac{a^2fg + s\omega_1(f - a - \omega_2)}{2s\omega_1 - 2ag\omega_2}.$$

We have the following possibilities:

Figure 1: Stability examples of Case 1 for P_1 . The values of the parameters used are $\omega_1 = \omega_2 = \frac{1}{2}$, $g = 10^4$, $s = 500$ and $f = 2/5$.



- (a) If $R < 0$, then the equilibrium point P_1 is unstable because the real part of one of the eigenvalues is positive.
- (b) If $R > 0$ and $\Delta < 0$, P_1 is asymptotically stable.
- (c) If $R > 0$ and $\Delta > 0$, the equilibrium point is asymptotically stable if

$$4a(f - \omega_2)(afg - s\omega_1)(ag\omega_2 - s\omega_1) < 0,$$

because both eigenvalues are negative if this inequality is satisfied.

- (d) If $R = 0$, P_1 is unstable if the following inequality holds:

$$4a(f - \omega_2)(afg - s\omega_1)(ag\omega_2 - s\omega_1) > 0.$$

Figure 1 illustrates an example of the change of stability of the equilibrium point P_1 depending on the value of the parameters considered. In Figure 1a, $R = \frac{1}{50} > 0$ and $\Delta = -1400 < 0$, thus P_1 is asymptotically stable, according to Case 1.c. The exact value of the fixed point is $(T^*, L^*) = (15000, 5000)$. In Figure 1b we have chosen $R = -\frac{17}{300} < 0$, and hence the equilibrium point $(T^*, L^*) = (27500, 15000)$ is unstable, as Case 1.a states.

2. **Case 2.** If the fixed point belongs to the line of fixed points L_2 in (8), then, the characteristic polynomial (11) has the form:

$$0 = \lambda^2(g + T)^2 + \lambda \left(-aT(g + T) + \frac{s(g + T)\omega_1}{a} \right),$$

and

$$\lambda_1 = 0, \quad \lambda_2 = \frac{a^2 T - s\omega_1}{ag(g + T)}.$$

The stability of the equilibrium points in L_2 cannot be set just with the study of the real part of the eigenvalues of the Jacobian matrix(10) since $\lambda_1 = 0$. But what can be determined is their instability region, depending on the sign of λ_2 .

Since the denominator of λ_2 is always positive we have:

$$\lambda_2 < 0 \Leftrightarrow a^2 T - s\omega_1 < 0 \Leftrightarrow T < \frac{s\omega_1}{a^2}.$$

Hence, if $T \in (\frac{s\omega_1}{a^2}, +\infty)$, the equilibrium points in L_2 are unstable.

3. **Case 3.** To finish our study, we analyze the tumor-free equilibrium point P_3 . The Jacobian matrix (10) takes the form:

$$J\left(0, \frac{s}{f}\right) = \begin{bmatrix} a - \frac{\omega_1 s}{f g} & 0 \\ \frac{\omega_2 s}{f g} & -f \end{bmatrix}.$$

The eigenvalues are the elements of the diagonal

$$\lambda_1 = -f, \quad \lambda_2 = a - \frac{\omega_1 s}{f g}.$$

The first one is always negative, thus the stability depends exclusively on the sign of λ_2 . So, P_2 is asymptotically stable if

$$a < \frac{\omega_1 s}{f g} \iff \frac{s}{f} > \frac{ag}{\omega_1}. \quad (13)$$

This condition determines a relationship between the tumor growth a and other values involved in the model, in particular, the lymphocyte production rate s and decay rate f .

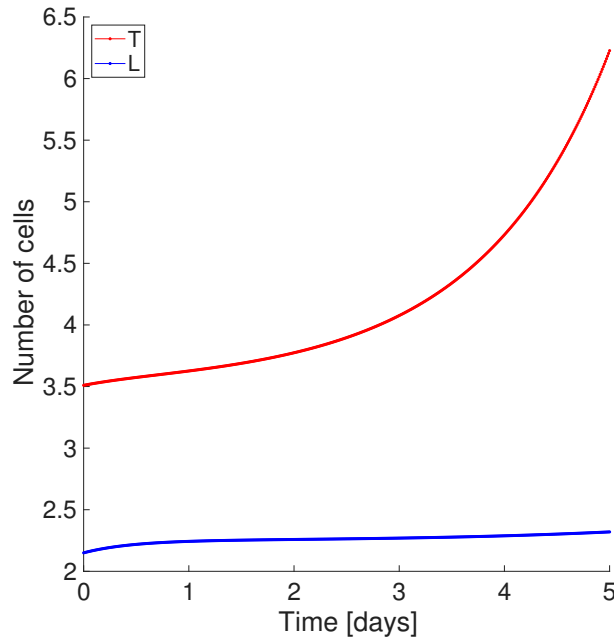
On the other hand, if the opposite inequality holds, the fixed point will not be stable.

Numerical Simulations

In this section some numerical experiments are performed in order to illustrate the behavior of the model (3) and confirm the results obtained in the previous section. To numerically solve the system, the fourth-order Runge-Kutta method has been used.

1. **Example 1.** We start with an example of an equilibrium point P_1 in Case 1. We set the values of the parameters as $\omega_1 = 2$, $\omega_2 = 0.1$, $a = 1$, $g = 1$, $s = 5$ and $f = 3$. With this parameters we obtain the equilibrium point $P_1 = (T^*, L^*) =$

Figure 2: Example 1. Approximate solution of an unstable equilibrium point of type P_1 . The parameters chosen in this example are $\omega_1 = 2, \omega_2 = 0.1, a = 1, g = 1, s = 5$ and $f = 3$ and initial values $T_0 = 3.51$ and $L_0 = 2.15$.



$(3.50, 2.25)$ which is unstable, since one of the eigenvalues of the Jacobian matrix of the system is positive ($\lambda_1 = 0.7190$ and $\lambda_2 = -2.1635$).

In Figure 2, the representation of the evolution with time of this example, with initial values $T_0 = 3.51$ and $L_0 = 2.15$, can be found. As it can be seen, the number of tumor cells quickly rises with time.

2. **Example 2.** We illustrate now the case of an equilibrium point of type $P_3 = (0, s/f)$. We set the parameters to $\omega_1 = 2, \omega_2 = 1, f = 3$ and $g = 1$.

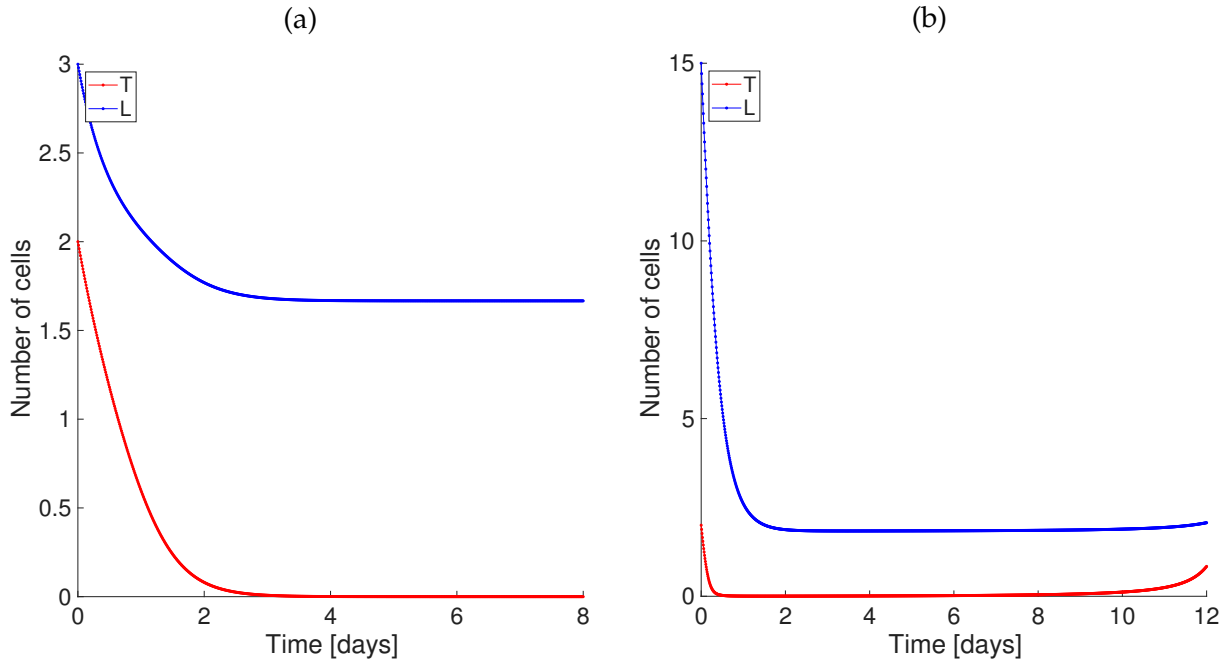
In Figure 3a we have an approximate solution obtained with $a = 1$ and $s = 5$. With these values, the inequality (13) (in this case $\frac{\omega_1 s}{f g} = 3.33 > 1 = a$), so, the corresponding point, $P_3 = (T^*, L^*) = (0, 5/3)$ is asymptotically stable, as it can be seen.

We increase the values of s and a so that (13) is not satisfied, for instance we choose $a = 4$ and $s = 5.5$. As it can be observed in Figure 3b, the equilibrium point $P_3 = (T^*, L^*) = (0, 11/6)$ is unstable. At the beginning of the simulation, one can see how the tumor cells decay almost to zero, but after approximately 10 days, they start to grow exponentially.

Conclusions

The model described in this work, adapted from [5], captures the interaction between tumor and immune system cells. The qualitative analysis performed showed that it

Figure 3: Example 2. Approximate solution of an stable (a) and unstable (b) equilibrium point of type P_3 . The parameters chosen in this example are $\omega_1 = 2, \omega_2 = 1, f = 3, g = 1$ and $a = 1, s = 5, T_0 = 2$ and $L_0 = 3$ in (a) and $a = 4, s = 5.5, T_0 = 2$ and $L_0 = 15$ in (b).



has three different types of equilibrium points and that their stability depends on some relations and inequalities between their constitutive parameters. The numerical examples validate the theoretical results obtained.

Although the model is a simplification of reality, it serves as a useful starting point to incorporate more variables, ultimately making the equations suitable to evaluate the results of different medical therapies.

Acknowledgements

All three authors have been supported by grant PID2023-146836NB-I00, funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU. A.B and M.C.M have been supported by GVA project CIAICO/2024/089.

References

- [1] Benzekry, S. et al. (2014). Classical Mathematical Models for Description and Prediction of Experimental Tumor Growth. PLoS Computational Biology, 10(8), e1003800.
- [2] Global Cancer Observatory. World Health Organization. <https://gco.iarc.who.int/en>

- [3] Monod, J. (1949). The Growth of Bacterial Cultures. *Annual Review of Microbiology*, 3, 371. doi:10.1146/annurev.mi.03.100149.002103
- [4] Sancho-Araiz, A. et al. (2021). The Role of Mathematical Models in Immunology: Challenges and Future Perspectives. *Pharmaceutics*, 13(7), 1016.
- [5] Sung, W. et al. (2020). A tumor-immune interaction model for hepatocellular carcinoma based on measured lymphocyte counts in patients undergoing radiotherapy. *Radiotherapy and Oncology*, 151, 73–81.

The Hidden Frequencies of Functions: The Mathematics of Harmonic Analysis

Irene Buffi ^{*1} and Gaia Mereu ^{†2}

¹*Dipartimento di Matematica, Università di Trento, Via Sommarive, 14, 38123–Trento, Italy*

²*Dipartimento di Matematica, Università degli Studi di Ferrara, Via Machiavelli, 30, 44121–Ferrara, Italy*

Introduction

Harmonic Analysis is a fundamental field in mathematics that provides the essential framework for understanding complex physical phenomena, most notably the nature of sound and human perception. The central objective of this paper is to leverage the mathematical tools of Harmonic Analysis to rigorously analyze a common auditory observation: **why do two different instruments playing the same musical note produce distinct acoustic qualities, or timbres?** A musical note is primarily defined by its **fundamental frequency** (for example A4 is 440 Hz). However, when a flute and a violin both play A4, the listener perceives two audibly separate sounds. This distinction cannot be explained solely by simple frequency analysis; it lies in the complex composition of overtones, known as **harmonics**. Mathematically, sound is the propagation of pressure waves, typically modeled by periodic functions in the time domain. The **sine function**, $f(t) = A \sin(\omega t)$ (Fig.1), serves as the canonical building block for sound analysis because it represents a single, pure frequency—the simplest possible tone. Real-world sounds, however, are rarely pure tones; they are intricate combinations of the fundamental frequency and numerous integer multiples of that frequency, known as the harmonic series.

The mathematical mechanism capable of deconstructing this complexity is **Fourier Analysis**. Developed from the work of Joseph Fourier, this analytical method posits that any sufficiently well-behaved periodic function can be uniquely represented as an infinite sum of simple sine and cosine waves—the Fourier Series. The coefficients of this series precisely quantify the amplitude and phase of each harmonic component, thereby defining the unique spectral fingerprint, or timbre, of a sound. [1] [2][3][4]

^{*}ibuffi@alumni.uv.es

[†]megai@alumni.uv.es

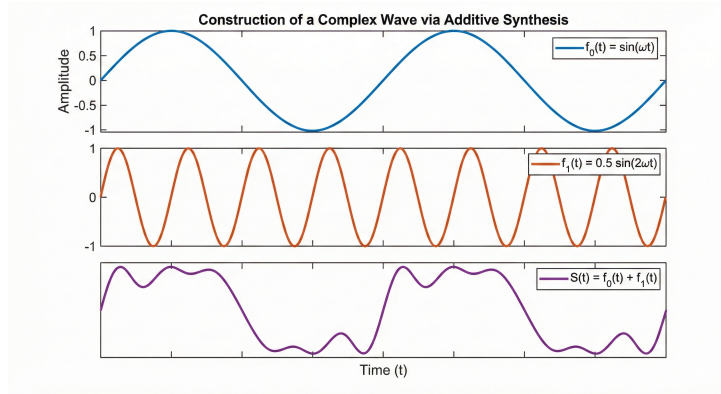


Figure 1: Construction of a Complex Wave

The Physics and Mathematics of Sound Perception

To rigorously analyze the harmonic structure of music, one must first establish a precise definition of the physical medium of sound and the biological mechanism of perception. This requires investigating the statistical mechanics of air, the anatomy of the cochlea, and the differential equations governing harmonic motion.

Sound is physically defined as the propagation of pressure waves through a medium, typically air. To understand this, we must model air not as a continuous fluid, but as a collection of discrete molecules in rapid motion. At room temperature, the mean velocity of air molecules is approximately 450–500 m/s. These molecules do not collapse under gravity due to their high kinetic energy and frequent collisions; the *mean free path* (the average distance a molecule travels before colliding) is roughly 6×10^{-8} meters, resulting in a collision frequency of approximately 10^{10} collisions per second. A sound wave is a local disturbance in this pressure field. It propagates at the speed of sound (approx. 340 m/s), which is distinct from the velocity of the individual molecules. Crucially, sound waves are **longitudinal**, meaning the oscillation of particles occurs parallel to the direction of propagation, unlike the transverse nature of electromagnetic or water waves. The perception of these waves relies on four main attributes: amplitude (loudness), frequency (pitch), spectrum (timbre), and duration. (Fig.2)

The Physiology of the Harmonic Analyzer: The Cochlea

The conversion of air pressure variations into electrical signals occurs in the ear, which acts as a sophisticated mechanical transducer. The outer ear channels sound to the *tympanic membrane* (eardrum), which transmits vibrations via the ossicular chain (hammer, anvil, and stirrup) to the *oval window* of the inner ear.

The core of auditory analysis lies in the **cochlea**, a fluid-filled, spiral cavity partitioned by the **basilar membrane**. This membrane is not uniform; it tapers in width and varies in stiffness along its length. Mathematically, this structure functions as a biological **Fourier Analyzer**. When a sound wave excites the fluid in the cochlea, it creates a traveling wave along the basilar membrane. Because the resonant frequency of the membrane varies with position (decreasing from the basal end to the apical end),

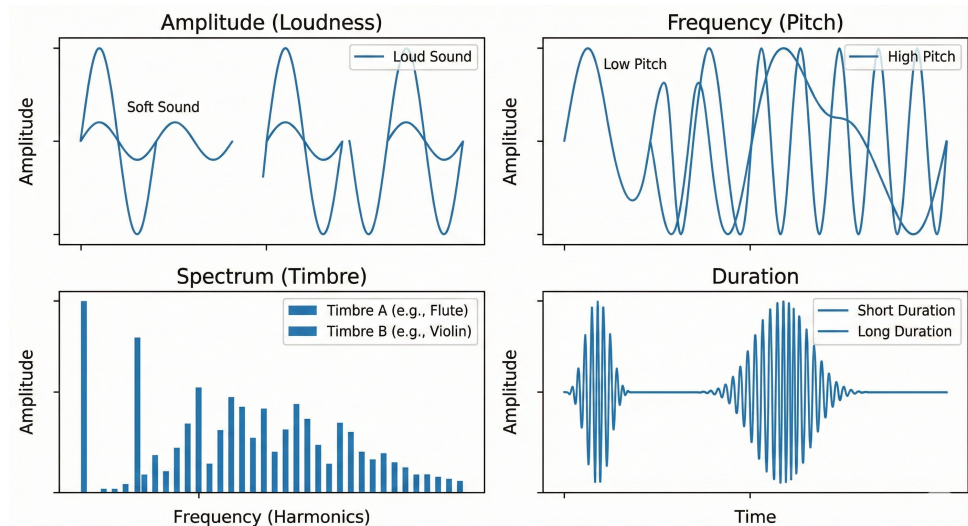


Figure 2: Amplitude, frequency, spectrum, duration

a specific frequency will cause a maximum amplitude of vibration at a specific “place” on the membrane.

This mechanism is formalized in two principles:

1. **Ohm’s Acoustic Law:** The ear perceives a complex sound by resolving it into its constituent sinusoidal frequency components.
2. **Helmholtz’s Place Theory:** The brain determines pitch based on which specific hair cells along the basilar membrane are stimulated.

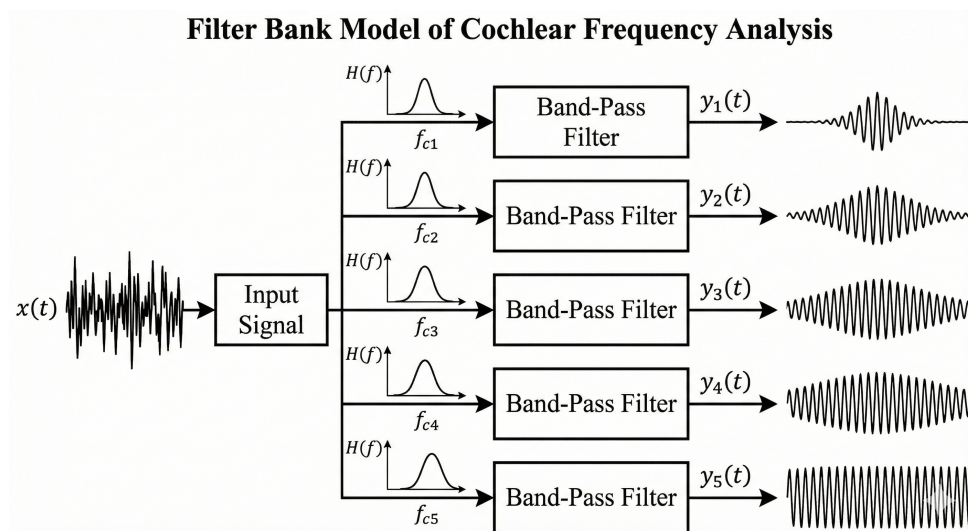


Figure 3: Model illustrating cochlear frequency analysis.

As you can see in Fig. 3, the complex input signal, $x(t)$, is processed in parallel by a series of band-pass filters, each tuned to a different center frequency ($f_{c1}, f_{c2}, \dots$), as indicated by their frequency responses $H(f)$. This decomposition results in multiple

narrow-band output signals $(y_1(t), y_2(t), \dots)$, effectively modeling the real-time spectral analysis performed mechanically by the inner ear.

Mathematical Derivation of the Sine Function

The justification for using the sine function ($f(t) = \sin t$) as the fundamental atom of sound analysis arises directly from the differential equation governing the basilar membrane's motion. We model a point on the membrane as a particle of mass m subject to a restoring force F proportional to its displacement y from equilibrium (Hooke's Law):

$$F = -ky \quad (1)$$

where k is the stiffness constant. By Newton's Second Law ($F = ma$), we obtain the equation of motion:

$$m \frac{d^2 y}{dt^2} + ky = 0 \quad \Rightarrow \quad \frac{d^2 y}{dt^2} + \frac{k}{m} y = 0 \quad (2)$$

This is the differential equation for **Simple Harmonic Motion**. Its general solutions are exclusively sinusoidal:

$$y(t) = A \cos \left(\sqrt{\frac{k}{m}} t \right) + B \sin \left(\sqrt{\frac{k}{m}} t \right) \quad (3)$$

Defining the angular frequency $\omega = \sqrt{k/m}$, this can be rewritten in terms of amplitude c and phase ϕ :

$$y(t) = c \sin(\omega t + \phi) \quad (4)$$

This derivation proves that the sine wave is not an arbitrary choice, but the **general solution** to the physical system of the ear. A complex musical note behaves as a superposition of these elementary solutions, leading to the necessity of Fourier Analysis.

Fourier Analysis

Mathematical formalism of Fourier Analysis

Before addressing any physical applications, it is necessary to define the rigorous mathematical framework within which we operate. Let $f : [0, T] \rightarrow \mathbb{R}$ be a Riemann integrable function. We define the real Fourier coefficients of f as:

$$\begin{aligned} a_0 &= \frac{2}{T} \int_0^T f(t) dt \\ a_n &= \frac{2}{T} \int_0^T f(t) \cos \left(\frac{2\pi n t}{T} \right) dt \\ b_n &= \frac{2}{T} \int_0^T f(t) \sin \left(\frac{2\pi n t}{T} \right) dt \end{aligned}$$

for $n \geq 1$. The associated Fourier series is given by

$$S_f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos\left(\frac{2\pi nt}{T}\right) + b_n \sin\left(\frac{2\pi nt}{T}\right) \right)$$

The assumption of Riemann integrability made earlier in this paper is required solely to ensure that the Fourier coefficients are well-defined. As will be shown later, assuming square integrability will suffice.

A central problem of Fourier analysis is to determine in which sense, and under which assumptions, the Fourier series converges to the original function. In this work, we focus on the convergence of Fourier series for L^2 functions, a space defined as follows. Let us consider the function space $L^2([0, T])$, consisting of the equivalence classes of real-valued measurable functions defined on the interval $[0, T]$ that are "square-integrable." Formally, a function f belongs to this space if:

$$\int_0^T |f(t)|^2 dt < \infty$$

We endow this space with an inner product, defined by

$$\langle f, g \rangle = \int_0^T f(t)g(t) dt$$

which naturally induces a norm $\|f\|_2 = \sqrt{\langle f, f \rangle}$. Upon establishing this metric structure, the space satisfies the axioms of a Hilbert space. The completeness of this space with respect to this norm guarantees that Cauchy sequences converge to a limit within the space itself, allowing the extension of Euclidean concepts, such as orthogonality, to infinite dimensions. Furthermore, we have that within $L^2([0, T])$, the set of trigonometric functions

$$\left\{ \cos\left(\frac{2\pi nt}{T}\right), \sin\left(\frac{2\pi nt}{T}\right) \right\}_{n \in \mathbb{N}}$$

forms a complete orthogonal basis.

In the Fourier analysis context, the core of L^2 theory lies in the manner in which a series approximates the original function. This behavior is formalized by the Mean Square Convergence Theorem. Denoting by

$$S_N(t) = \frac{a_0}{2} + \sum_{n=1}^N \left(a_n \cos\left(\frac{2\pi nt}{T}\right) + b_n \sin\left(\frac{2\pi nt}{T}\right) \right)$$

the N -th partial sum of the Fourier series of f , the theorem states that for every $f \in L^2([0, T])$,

$$\lim_{N \rightarrow \infty} \|f - S_N\|_2 = \lim_{N \rightarrow \infty} \left(\int_0^T |f(t) - S_N(t)|^2 dt \right)^{1/2} = 0$$

This means that the Fourier series of f converges to f in the L^2 -norm.

From a physical perspective, this result admits a natural interpretation: as the number of terms in the series increases, the energy of the approximation error, measured in the mean square sense, tends to zero. The partial sums therefore provide the best possible approximation to f among all trigonometric polynomials of the same order.

However, convergence in energy does not imply pointwise or uniform perfection. In particular, when f exhibits jump discontinuities, the partial sums of its Fourier series display persistent oscillations in any neighborhood of such points. This behavior, known as the Gibbs phenomenon, is characterized by an overshoot whose maximum does not vanish as $N \rightarrow \infty$, but instead converges to a fixed proportion of the jump size (approximately 9%). Although these oscillations become increasingly localized, uniform convergence fails in the presence of discontinuities, revealing an intrinsic limitation of representing non-smooth functions using smooth, global basis functions.

A fundamental consequence of the completeness of the trigonometric system in $L^2([0, T])$ is Parseval's identity, which establishes an isometric correspondence between a function and its Fourier coefficients:

$$\|f\|_2^2 = T|a_0|^2 + \frac{T}{2} \sum_{n=1}^{\infty} (|a_n|^2 + |b_n|^2)$$

This identity endows Fourier analysis with a powerful alternative representation of functions, commonly known as the frequency domain. Rather than describing a phenomenon through its temporal evolution $f(t)$, one may characterize it entirely through its spectral content, using the coefficients $\{a_n, b_n\}$. This paradigm shift is essential for acoustics: what appears mathematically as a series decomposition physically corresponds to the mechanism by which the ear and brain decode the timbre of musical sounds.

Fourier Analysis in the modeling of sound

With the fundamental mathematical framework in place, we now turn to a detailed examination of its application to the analysis of sound signals. As outlined in the preceding section, sound is physically defined as a pressure perturbation propagating through an elastic medium, such as air, in the form of waves. A sustained musical sound, such as a steady note, can be modeled as a time-dependent pressure signal p that is approximately periodic over sufficiently short time intervals. More precisely, there exists a period $T > 0$ such that $p(t + T) \approx p(t)$. This approximation accounts for the presence of slow amplitude variations and other non-ideal effects inherent in real sounds. Such near-periodicity justifies restricting the analysis to a finite time window $[0, T]$, allowing the use of the Fourier series as an effective local representation of the signal.

A fundamental physical constraint is that the total energy carried by a sound must be finite, as no real sound signal propagates indefinitely. Letting p denote the acoustic pressure variation, the instantaneous acoustic energy density is proportional to $p(t)^2$. Therefore, the total energy of the signal is proportional to the integral over time of the

squared pressure amplitude, that is

$$E_{\text{total}} \propto \int_{-\infty}^{+\infty} |p(t)|^2 dt$$

Every physically realizable sound must necessarily satisfy the condition of having finite total energy. Consequently, real sound signal are naturally modeled as functions belonging to the space $L^2(\mathbb{R})$, that is, functions $f: \mathbb{R} \rightarrow \mathbb{R}$ such that:

$$\int_{-\infty}^{+\infty} |f(t)|^2 dt < \infty$$

This condition provides the appropriate mathematical expression of the physical requirement that no sound can propagate indefinitely with nonzero energy. In practical and perceptual contexts, however, sound signals are never observed over an infinite time interval. Both physical measurement devices and the human auditory system operate on finite time windows, during which the signal can be regarded as approximately stationary. For this reason, it is natural to restrict the analysis to a finite interval $[0, T]$, corresponding to a temporal window over which the sound is assumed to be locally periodic or statistically stable. The restriction of a function in $L^2(\mathbb{R})$ to such a finite interval defines an element of $L^2([0, T])$. This transition from $L^2(\mathbb{R})$ to $L^2([0, T])$ allows the use of Fourier series as a discrete spectral representation, while retaining the physically meaningful interpretation of energy. Moreover, the reliance on square integrability rather than on stronger regularity assumptions, such as pointwise continuity, reflects the intrinsic nature of acoustic signals. Real sounds frequently exhibit irregular features, including transients and noise, for which pointwise continuity is neither physically justified nor perceptually relevant. Instead, human auditory perception is primarily sensitive to the distribution of energy across frequencies, a property that is naturally captured within the L^2 framework.

Given this premise, let us consider an acoustic signal $p(t)$ defined on a finite time interval $[0, T]$ and specifically its Fourier series decomposition:

$$p(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos\left(\frac{2\pi nt}{T}\right) + b_n \sin\left(\frac{2\pi nt}{T}\right) \right)$$

In this formula, the coefficient a_0 is the constant component (DC) of the signal and is equal to twice the average value of p over the interval

$$\frac{a_0}{2} = \frac{1}{T} \int_0^T p(t) dt$$

Physically, this term corresponds to a constant average pressure offset. Since it is not associated with any oscillatory behavior, it does not produce an auditory sensation and therefore does not contribute to perceived pitch, loudness, or timbre. For this reason, and without loss of generality, the constant component may be neglected in the perceptual analysis of sound. Defining the fundamental frequency as $f_0 = \frac{1}{T}$, the

remaining components with frequency $f_n = nf_0$ are called harmonics. From a physical point of view, it is useful to rewrite the Fourier series by combining sine and cosine into a single oscillation with amplitude and phase as follows:

$$p(t) = \sum_{n=1}^{\infty} A_n \cos(2\pi n f_0 t + \phi_n)$$

In this expression, $A_n = \sqrt{a_n^2 + b_n^2}$ denotes the amplitude of the n -th harmonic, while $\phi_n = \arctan\left(-\frac{b_n}{a_n}\right)$ its phase. From a mathematical perspective, both amplitude and phase are required for the exact reconstruction of the signal. However, psychoacoustic studies indicate that, for steady-state sounds, human auditory perception is largely insensitive to relative phase information. According to Ohm's acoustic law, the auditory system acts primarily as a frequency analyzer, responding to the magnitudes of spectral components rather than to their phases. Consequently, acoustic signals sharing the same amplitude spectrum $\{A_n\}$ are perceived as having the same timbre, even in the presence of different phase configurations. This perceptual characteristic justifies the omission of phase terms in our model, thereby restricting the characterization of timbre to the distribution of spectral energy, denoted here by $\mathcal{S}(p)$.

From a physical point of view, this decomposition is further motivated by the fact that sinusoidal functions are elementary solutions of the one-dimensional wave equation $\partial_{tt}u = c^2\partial_{xx}u$. Therefore, any complex oscillation can be interpreted as a linear superposition of such fundamental modes.

The total energy of the signal in the considered interval is given by:

$$E = \int_0^T |p(t)|^2 dt$$

By Parseval's theorem, and under the assumption that the constant component has been neglected, this energy can be expressed purely in terms of the harmonic amplitudes as

$$E = \frac{T}{2} \sum_{n=1}^{\infty} A_n^2$$

This identity provides a direct link between the physical notion of energy and the spectral representation of the signal, reinforcing the central role of Fourier analysis in the modeling and perception of sound.

Two sounds sharing the same fundamental frequency f_0 but characterized by different sets of harmonic amplitudes $\{A_n\}$ are mathematically distinct elements in $L^2([0, T])$ and are physically perceived as the same musical note with different timbres. In the analysis of periodic sound signals, it is crucial to distinguish between two primary perceptual attributes that find a precise mathematical interpretation: pitch and timbre. Pitch is uniquely determined by the fundamental frequency f_0 of the signal $p(t)$, while timbre depends on the internal spectral composition of the signal itself. More precisely, given two signals $p_1(t)$ and $p_2(t)$ in the space $L^2([0, T])$, they possess the same pitch if

they share the same fundamental period T , and thus the same fundamental frequency f_0 . However, this condition does not imply that the signals coincide. We formally define the harmonic spectrum of a signal p as the ordered set of the amplitudes of its harmonic components,

$$\mathcal{S}(p) = \{A_n\}_{n \in \mathbb{N}}$$

where $A_n = \sqrt{a_n^2 + b_n^2}$ denotes the amplitude of the n -th harmonic associated with the frequency nf_0 . Timbre is determined by the manner in which the signal's energy is distributed among these harmonics, and is therefore intrinsically linked to the structure of the spectrum $\mathcal{S}(p)$. Mathematically, the completeness and orthogonality of the trigonometric basis in $L^2([0, T])$ guarantee that a signal is uniquely characterized (up to equivalence in L^2) by its Fourier coefficients. Consequently, two sounds with identical fundamental frequency are perceptually indistinguishable only if their harmonic spectra coincide. Conversely, any difference in their spectral composition, even in a single harmonic component, results in distinct elements of $L^2([0, T])$ and give rise to a perceptible difference in timbre. Formally, denoting by $\{A_n^{(1)}\}$ and $\{A_n^{(2)}\}$ the spectra of two signals p_1 and p_2 respectively, we have:

$$\exists n \in \mathbb{N} : A_n^{(1)} \neq A_n^{(2)} \Rightarrow p_1 \neq p_2$$

This criterion provides a precise mathematical formulation of timbral distinction, interpreting timbre as a spectral fingerprint of the sound. In the following section, this abstract characterization will be illustrated through concrete examples, showing how different spectral distributions correspond to the characteristic timbres of distinct musical sounds.

Spectral Characterization of Canonical Waveforms

While the mathematical definition of timbre relies on the uniqueness of the Fourier coefficient set $\mathcal{S}(p)$, it is physically instructive to examine the spectral decomposition of idealized waveforms that serve as canonical models for acoustic instruments. By explicitly computing the Fourier series for these functions, one can directly observe how the mathematical structure of harmonic spectrum shape auditory perception. In what follow, we consider two fundamental waveforms: the sawtooth wave and the square wave. The sawtooth wave is the fundamental model for the oscillation produced by bowed string instruments, such as the violin or cello, as well as brass instruments, where sound production is governed by nonlinear excitation mechanisms. Consider the function $f(t) = t$ defined on the interval $[-\pi, \pi]$ and extended periodically. Its Fourier series expansion is given by

$$f(t) = 2 \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n} \sin(nt)$$

An analysis of the corresponding spectral amplitudes A_n shows that the magnitude of the n -th harmonic is proportional to the inverse of its harmonic number:

$$A_n \propto \frac{1}{n}$$

The summation runs over all integers $n \in \mathbb{N}$, implying that the spectrum contains both odd and even harmonics of the fundamental frequency. This dense harmonic structure results in a spectrally rich sound, characterized by significant high-frequency content. Perceptually, such a spectrum is associated with brightness and brilliance, qualities commonly attributed to string sections and brass instruments in orchestral contexts.

In contrast, the square wave serves as a model for the acoustic behavior of cylindrical air columns closed at one end, such as the clarinet in its lower register. Defined as a function that alternates between distinct values (e.g., 1 for $t \in [0, \pi]$ and -1 for $t \in [\pi, 2\pi]$), its Fourier series is given by:

$$f(t) = \frac{4}{\pi} \sum_{k=1}^{\infty} \frac{\sin((2k-1)t)}{2k-1}$$

In this case, only odd harmonics appear in the expansion. Accordingly, the spectral amplitudes satisfy

$$A_n \propto \begin{cases} \frac{1}{n} & \text{if } n \text{ is odd} \\ 0 & \text{if } n \text{ is even} \end{cases}$$

This mathematical result has a direct acoustic consequence: the square wave exhibits systematic spectral gaps at all even integer multiples of the fundamental frequency. This suppression of even harmonics is responsible for the characteristic hollow, woody, or nasal timbre commonly associated with the clarinet. This example illustrates that timbre is not determined solely by the presence of overtones, but equally by their structured absence.

Together, these canonical examples demonstrate how distinct harmonic organizations, despite sharing the same fundamental frequency, give rise to markedly different perceptual qualities. They provide concrete evidence that timbre is intrinsically encoded in the spectral distribution of energy, thereby reinforcing the central role of Fourier analysis in the mathematical modeling of sound.

Applications: Mathematical Synthesis and Physical Reality

While Fourier Theory provides the analytical tools to decompose periodic functions, its application to sound synthesis requires addressing the dynamic and non-ideal nature of physical reality. This section formalizes the mathematical models used to reconstruct complex timbres, moving from the discrete Fourier coefficients to the continuous sinusoidal representation used in modern signal processing.

The Sinusoidal Model and Time-Varying Spectra

The classical Fourier Series assumes strict periodicity, implying that coefficients a_n and b_n are constant over time. However, natural sounds exhibit time-varying spectral characteristics. To model this, we extend the Fourier definition to the **Sinusoidal Model**

proposed by McAulay and Quatieri. The signal $s(t)$ is modeled as a sum of $L(t)$ time-varying sinusoids:

$$s(t) = \sum_{l=1}^{L(t)} A_l(t) \cos(\theta_l(t)) \quad (5)$$

where $A_l(t)$ is the time-varying amplitude envelope and $\theta_l(t)$ is the total phase. The instantaneous frequency $\omega_l(t)$ is defined as the derivative of the phase:

$$\omega_l(t) = \frac{d\theta_l(t)}{dt} \quad (6)$$

This formulation allows for the "birth" and "death" of harmonic tracks, where $L(t)$ changes as partials enter or leave the audible spectrum.

Cubic Phase Interpolation

A critical mathematical challenge in additive synthesis is reconstructing a continuous phase function $\theta(t)$ from discrete Short-Time Fourier Transform (STFT) frames. Given a frequency match between frame k and frame $k + 1$, simple linear interpolation of frequency fails to preserve phase continuity, resulting in audible artifacts.

To ensure "maximal smoothness," the phase function is modeled as a **cubic polynomial** in time:

$$\theta(t) = \zeta + \gamma t + \alpha t^2 + \beta t^3 \quad (7)$$

We must solve for the coefficients $(\zeta, \gamma, \alpha, \beta)$ subject to four boundary conditions derived from the STFT analysis at frame boundaries $t = 0$ and $t = T$:

$$\theta(0) = \theta_k \quad \theta(T) = \theta_{k+1} + 2\pi M \quad (8)$$

$$\dot{\theta}(0) = \omega_k \quad \dot{\theta}(T) = \omega_{k+1} \quad (9)$$

Here, θ_k and ω_k are the measured phase and frequency at frame k . The integer M represents the phase unwrapping factor, chosen to minimize the curvature of the phase track (i.e., minimizing $\int_0^T [\ddot{\theta}(t)]^2 dt$).

Solving this system of linear equations yields the parameters for the cubic trajectory:

$$\begin{bmatrix} \alpha(M) \\ \beta(M) \end{bmatrix} = \begin{bmatrix} \frac{3}{T^2} & -\frac{1}{T} \\ -\frac{2}{T^3} & \frac{1}{T^2} \end{bmatrix} \begin{bmatrix} \theta_{k+1} - \theta_k - \omega_k T + 2\pi M \\ \omega_{k+1} - \omega_k \end{bmatrix} \quad (10)$$

As you can see in 4 illustrates the smooth cubic polynomial phase path, $\theta(t)$, connecting two synthesis frames over the time interval $[0, T]$. The trajectory is constrained

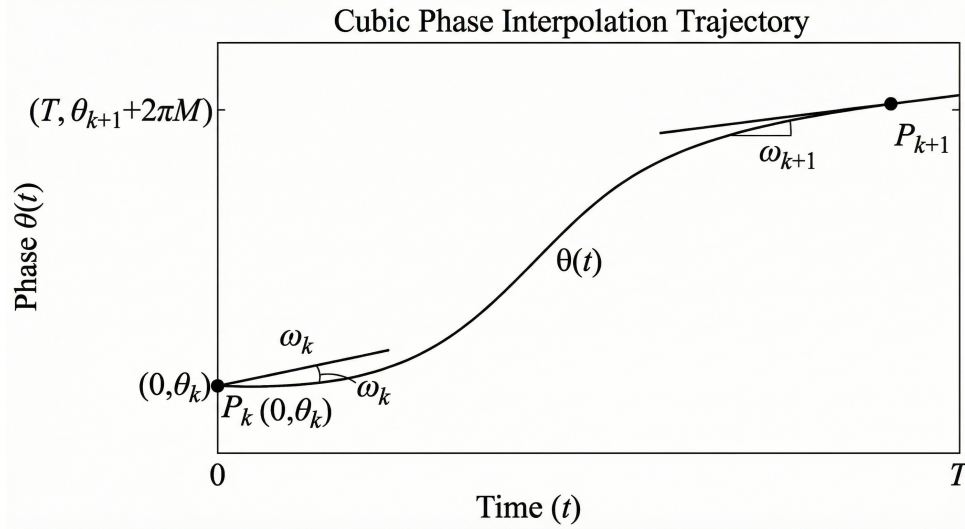


Figure 4: Cubic Phase Interpolation

by boundary conditions specifying both the exact phase values at the endpoints (P_k and P_{k+1}) and the required instantaneous frequencies (represented by the tangent slopes ω_k and ω_{k+1}), ensuring continuous differentiability across frame boundaries to prevent audible artifacts.

This rigorous interpolation ensures that the synthesized frequency track is continuously differentiable, preserving the harmonic integrity of the instrument's timbre.

Physical Inharmonicity: The Stiff String Equation

Finally, we must account for deviations from the ideal Fourier series caused by physical stiffness. In ideal Harmonic Analysis, partials are integer multiples ($f_n = n f_0$). However, for a physical string with non-negligible stiffness (such as a piano string), the restoring force includes a bending moment proportional to the fourth derivative of displacement. The wave equation becomes a fourth-order Partial Differential Equation:

$$\frac{\partial^2 y}{\partial t^2} = c^2 \frac{\partial^2 y}{\partial x^2} - \kappa^2 c^2 \frac{\partial^4 y}{\partial x^4} \quad (11)$$

where κ is the radius of gyration. The eigensolutions to this equation yield an **inharmonic spectrum**, where the n -th partial frequency is given by:

$$f_n = n f_0 \sqrt{1 + B n^2} \quad (12)$$

where B is the inharmonicity coefficient. This mathematical result explains why the harmonic series derived in Section 3 is an idealization, and why synthesis algorithms must stretch the harmonic series to replicate realistic acoustic instruments.

Conclusion: Temporal Limits and Microsound

In this work, we have analyzed the profound interconnection between musical perception and mathematics, demonstrating how timbre can be rigorously defined through its spectral signature. We have traced the trajectory from the physics of pressure waves (Section 2), through the theory of Fourier Series (Section 3), to its application in sinusoidal synthesis (Section 4).

However, it is fundamental to conclude by highlighting the intrinsic limits of classical Fourier Analysis. The sinusoidal model assumes that the signal is *stationary* or that it varies slowly over time. Musical reality, conversely, is rich in rapid transient events (such as the strike of a hammer or the pluck of a string) that evade traditional spectral analysis.

As discussed by Curtis Roads in his analysis of "Microsound", there exists a lower temporal scale (the micro scale, < 100 ms) where the perception of "pitch" disintegrates and the concept of frequency itself becomes ambiguous. In this domain, global harmonic analysis is insufficient because it fails to localize energy in time with sufficient precision. Roads highlights how the mathematical representation must therefore evolve from infinite Fourier series toward models based on time-frequency "atoms" (as seen in Wavelet or Granular analysis), acknowledging that sound is not merely a continuous spectral phenomenon, but a flow of discrete sonic particles.

In conclusion, while Fourier's Harmonic Analysis remains the fundamental alphabet for understanding tonality and macroscopic timbre, the temporal complexity of music requires a mathematical grammar that accepts the trade-off between frequency precision and temporal localization.

References

- [1] D. Benson. Music: a mathematical offering. Cambridge, Cambridge University Press, 2007.
- [2] E. M. Stein and R. Shakarchi. Fourier analysis. An Introduction. Princeton, NJ, Princeton University Press, 2003.
- [3] X. Serra and J. Smith III. Spectral Modeling Synthesis: A Sound Analysis Synthesis System Based on a Deterministic plus Stochastic Decomposition. *Computer Music Journal*, 1990.
- [4] C. Roads. Microsound. Cambridge, MA, The MIT Press, 2001.

Three unified Farey–sequence applications for rational direction discretization in digital image analysis

Harold Díaz-Iza^{*1,2}, Bushra Rani^{1,2}, and Ana Sáez-Morillo^{1,2}

¹*Universitat de València, Av. Vicent Andrés Estellés 19, 46100–Burjassot, Spain*

²*Universitat Politècnica de València, Camí de Vera s/n, 46022–València, Spain*

Abstract

In computer vision images are represented on an integer lattice $\mathbb{Z}^2$. However, many geometric primitives, such as lines, segments, and contours, are fundamentally continuous. In this study, we analyze three Farey-sequence applications that discretize these geometric primitives into a family of rational primitives with the aim of: (A1) detecting the longest digital straight segment in binary images; (A2) estimating slope for detected straight structures via Farey-based ordering and comparison of rational directions; and (A3) detecting lines based on Farey-guided rational Hough transform. These applications address common challenges in computer vision by effectively leveraging the unique properties of Farey sequences.

Introduction

In computer vision, images are represented on an integer lattice $\mathbb{Z}^2$. However, many geometric primitives, such as lines, segments, and contours, are fundamentally continuous [8, 14]. Bridging this mismatch is not merely a discretization detail: it directly impacts robustness, computational budget, and the reproducibility of the algorithms used in image analysis [9, 6].

A common practice of discretization of geometric primitives is to quantize orientations by uniform sampling in the angular orientation θ (angle binning), then rely on floating-point trigonometry and ad hoc neighborhood rules [17]. Moreover, when resources are limited, one needs a discretization whose resolution control is interpretable and deterministic, rather than a tuning of angular orientation step sizes [4].

This paper adopts an arithmetic viewpoint: when orientations are inferred from

^{*}hadiaiz@upvnet.upv.es

pixel coordinates or discrete contours, directions can be represented by *primitive* integer vectors (a, b) with $\gcd(a, b) = 1$, or equivalently by reduced rational slopes $\frac{b}{a}$ [10]. This perspective naturally leads to a classical object from number theory, the Farey sequence $\mathcal{F}_n$, which enumerates reduced fractions with denominators bounded by n in sorted order [13, 16]. Beyond providing a finite dictionary of rational directions, $\mathcal{F}_n$ equips that dictionary with an algorithmically functional structure: explicit neighbor relations, mediant-based refinement, and exact ordering, enabling quantization of multiresolution orientation without resorting to floating-point arithmetic.

In this paper, we cover three applications related with line and edge detection, where Farey-sequence is the common backbone for these computer-vision tasks:

- **Longest digital straight segment (DSS) detection in binary images:** we exploit the correspondence between Farey fractions and digitally realizable straight segments to detect, among candidate alignments, the longest consistent segment [3].
- **Exact slope identification and comparison:** we support reliable slope processing by addressing the *rank* and *order* queries in $\mathcal{F}_n$ via Farey tables, enabling exact comparisons and neighborhood-aware operations in a rational direction dictionary [4].
- **Farey-induced rational Hough transform:** for noisy edge maps, we discretize the orientation component of Hough by primitive integer normals induced by $\mathcal{F}_n$ and accumulate votes in the Diophantine parametrization $ax + by = s$, with resolution controlled explicitly by the Farey order n [5].

Two practical benefits can be observed through a single mathematical perspective. First, both resolution and control become clearer: the Farey order n dictates both the number of directions in the set and the complexity of the denominators for the slopes [11]. Second, essential subroutines become integer-friendly and accurate when applicable: processes such as slope ordering, indexing, and accumulator updates can be performed without trigonometric calculations and without relying on floating-point comparisons [12].

The remainder of the paper is organized as follows. Section introduce the mathematical background. Section presents the three applications built. Section reports the numerical results obtained in the three application. Section discusses the findings, including implications and limitations. Finally, Section presents the conclusion of the work.

Mathematical background

This section introduces the mathematical tools used throughout the paper. We first introduce notation and digital straightness primitives on $\mathbb{Z}^2$. We then define Farey sequences and summarize the properties used later: ordering, neighbor relations, and mediant refinement. Finally, we explain how a Farey dictionary defined on slopes in $[0, 1]$ induces a full 2D orientation discretization and how primitive integer normals yield an integer-friendly line parametrization connected to the classical Hough transform.

Notation and primitive integer directions

We work on the integer lattice $\mathbb{Z}^2$. A vector $(a, b) \in \mathbb{Z}^2$ is *primitive* if $\gcd(a, b) = 1$. Any nonzero displacement $(\Delta x, \Delta y) \in \mathbb{Z}^2$ reduces uniquely to a primitive direction

$$(\Delta x, \Delta y) \mapsto \left(\frac{\Delta x}{g}, \frac{\Delta y}{g} \right), \quad g = \gcd(\Delta x, \Delta y).$$

When $\Delta x \neq 0$, the corresponding slope is the reduced rational $m = \frac{\Delta y}{\Delta x}$; the vertical case $\Delta x = 0$ is treated separately.

Farey sequences: definition, size, and neighbors

The Farey sequence of order n , denoted $\mathcal{F}_n$, is the increasing list of reduced fractions $\frac{a}{b} \in [0, 1]$ with $0 \leq a \leq b \leq n$ and $\gcd(a, b) = 1$. Its cardinality satisfies

$$|\mathcal{F}_n| = 1 + \sum_{m=1}^n \varphi(m), \quad \text{and} \quad |\mathcal{F}_n| \sim \frac{3}{\pi^2} n^2, \quad (1)$$

where φ is Euler's totient function. If $\frac{a}{b} < \frac{c}{d}$ are consecutive terms in $\mathcal{F}_n$, then

$$bc - ad = 1, \quad \text{hence} \quad \frac{c}{d} - \frac{a}{b} = \frac{1}{bd}. \quad (2)$$

This yields an explicit neighbor structure in slope space.

Mediant and hierarchical refinement

Definition 1

[Mediant] For $\frac{a}{b}$ and $\frac{c}{d}$ with $b, d > 0$, their mediant is defined as follows

$$\frac{a}{b} \oplus \frac{c}{d} := \frac{a+c}{b+d}.$$

Mediants provide a refinement mechanism: new terms appearing at higher Farey orders arise as mediants of neighboring fractions:

Proposition 1

We can construct $\mathcal{F}_{n+1}$ from $\mathcal{F}_n$ as follows. For each pair of successive terms in $\mathcal{F}_n$ we consider their mediant $\frac{c}{d}$. If $d = n+1$ we insert it, otherwise we skip it.

Approximation by bounded-denominator rationals

For any real $x \in [0, 1]$ there exists a fraction $\frac{a}{b} \in \mathcal{F}_n$ such that

$$\left| x - \frac{a}{b} \right| < \frac{1}{b(n+1)}. \quad (3)$$

This motivates using n as an explicit resolution knob when quantizing real-valued slope estimates to a nearby rational direction.

From $\mathcal{F}_n$ on $[0, 1]$ to a full 2D orientation dictionary

Although $\mathcal{F}_n \subset [0, 1]$, it can be extended to all orientations using sign symmetries and the reciprocal transform $m \mapsto \frac{1}{m}$, with a separate convention for vertical directions ($m = \infty$). An equivalent representation uses primitive integer direction vectors (a, b) whose complexity is bounded by the Farey order. The parameter n thus controls both the admissible direction complexity and the resolution of orientation sampling.

Integer normal form and relation to the Hough transform

A line can be represented in the integer normal form

$$ax + by = s, \quad (a, b) \in \mathbb{Z}^2, \gcd(a, b) = 1, s \in \mathbb{Z}, \quad (4)$$

where (a, b) is a primitive integer normal and s is an integer offset. For comparison with the classical normal form $\rho = x \cos \theta + y \sin \theta$, one may associate

$$\theta = \text{atan2}(b, a), \quad \rho = \frac{s}{\sqrt{a^2 + b^2}}. \quad (5)$$

Hence, an accumulator indexed by (a, b, s) can be interpreted as a rationally sampled variant of (θ, ρ) space, where the set of admissible orientations is induced by primitive integer arithmetic rather than uniform angular bins [1, 5].

Three applications built on the same primitives

This section develops the three mentioned applications of Farey sequences in computer vision. We present: (A1) longest digital straight segments through a pixel, (A2) slope determination and contour processing via Farey ranks, and (A3) a Farey-discretized rational Hough transform.

A1: Longest digital straight segments through a given pixel

Problem statement

Given a binary image and a pixel p in it, our goal is to find (i) the longest white rays starting from p and (ii) the longest white segments centered at p . The methods should not be merely approximate, but report the exact answer regardless of the image.

Foundations

A proof of the correspondence between Farey sequences and digital straight lines can be found in [15]. In this section, we supply the details omitted in that proof.

Definition 2

A naive digital straight line, denoted by $D(a, b, \mu)$, with $a, b, \mu \in \mathbb{Z}$ and $\gcd(a, b) = 1$, is the set of points $(x, y) \in \mathbb{Z}^2$ verifying

$$\mu \leq ax - by < \mu + \max(|a|, |b|).$$

We restrict ourselves to naive lines with $\mu = 0$ and $0 \leq a \leq b$. These lines live in the first and fifth octants. In this case the a naive digital straight line is the set of points $(x, y) \in \mathbb{Z}^2$ verifying $y = \lfloor \frac{a}{b}x \rfloor$, which motivates the following definition:

Definition 3

We write, for $r \in [0, 1]$, $D(r) := \{(x, y) \in \mathbb{Z}^2 : y = \lfloor rx \rfloor\}$.

Definition 4

We denote by $G_n^I := \{(x, y) \in \mathbb{Z}^2 : 0 \leq x \leq n, 0 \leq y \leq x\}$ the set of integer points with bounded coordinates in the first octant.

Proposition 2

If r is any real number such that $\frac{a}{b} < r < \frac{a'}{b'}$, with $\frac{a}{b}, \frac{a'}{b'}$ consecutive in $\mathcal{F}_n$, then $D(\frac{a}{b}) \cap G_n^I = D(r) \cap G_n^I$

Proof

For each $0 \leq x \leq n$, both sets contain exactly one point, (x, y) and (x, z) , given by $y = \lfloor \frac{a}{b}x \rfloor$ and $z = \lfloor rx \rfloor$, respectively. Fix x . For $x = 0$ both sets contain the point $(0, 0)$. Now suppose $x > 0$. Clearly $y \leq z$. Suppose we have $y < z$. On the one hand, since $\lfloor \frac{a}{b}x \rfloor < \lfloor rx \rfloor$ implies that $\frac{a}{b}x < z$, we obtain that $\frac{a}{b} < \frac{z}{x}$. On the other hand, $\frac{z}{x} \leq r < \frac{a'}{b'}$. Thus $\frac{z}{x}$ is a term in $\mathcal{F}_n$ strictly between two successive terms, which is impossible. $\square$

As a direct consequence we obtain:

Proposition 3

We have that

$$S_n^I := \left\{ D\left(\frac{a}{b}\right) \cap G_n^I : \frac{a}{b} \in F_n \right\} = \{ D(r) \cap G_n^I : r \in [0, 1] \}.$$

Proposition 4

There is a bijection

$$\varphi : \mathcal{F}_n \rightarrow S_n^I, \quad \varphi\left(\frac{a}{b}\right) = D\left(\frac{a}{b}\right) \cap G_n^I.$$

Proof

That φ is surjective follows directly from previous proposition. To show that is injective, let $\frac{a}{b}, \frac{c}{d} \in F_n$ with $\frac{a}{b} < \frac{c}{d}$, and we'll see that $\varphi(\frac{a}{b}) \neq \varphi(\frac{c}{d})$ proving that the point $(d, c) \in \varphi(\frac{c}{d})$ is not in $\varphi(\frac{a}{b})$. If it were, we would have

$$c = \left\lfloor \frac{a}{b}d \right\rfloor \implies c \leq \frac{a}{b}d \implies \frac{c}{d} \leq \frac{a}{b}.$$

$\square$

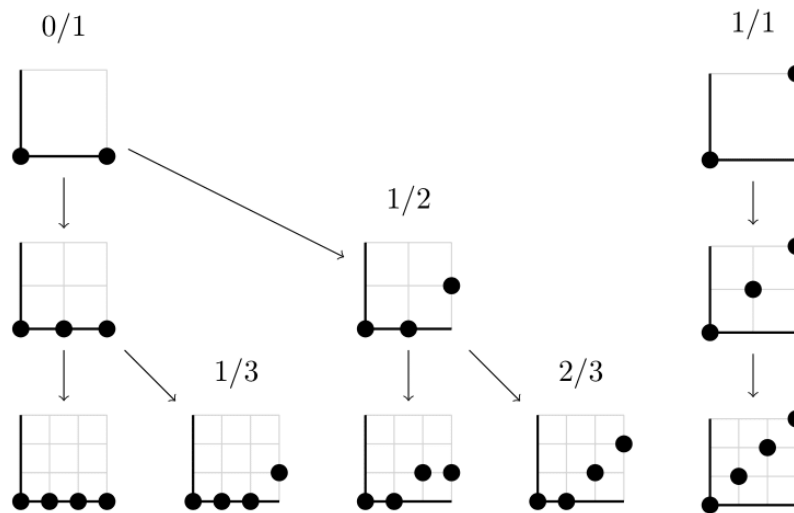


Figure 1: Correspondence between naive straight segments in the first octant and slopes in the Farey sequence. The first, second and third rows correspond to Farey sequences of order 1, 2 and 3, respectively. For $n = 1, 2, 3$, the n -th row contains all elements of the set S_n^I of Proposition 3.

Proposed method for rays

The method is described in Algorithm 1 and can be summarized as follows. Let p be a given pixel of a binary image I . Our goal is to find the longest naive digital straight segment starting from p , or *rays*, consisting only of white pixels. Without loss of generality, we can assume that the segments are in the first octant, so they correspond to slopes between 0 and 1. Other octants can be handled by symmetry, as summarized in Section .

The algorithm proceeds incrementally. For each n , T_n is the set of slopes under consideration for rays of length n , and B_n is a boolean vector, whose i -th term is a one if and only if the ray corresponding to the i -th slope in T_n is white. In the n -th step, the algorithm computes the vector T_{n+1} (T' in Algorithm 1) from T_n and B_n as follows. We iterate through T_n using the enumerator k . If $B_n(k) = 1$ or $B_n(k-1) = 1$, we add $T_n(k)$ to T_{n+1} . In the case $T_n(k)$ is not the last element, we consider the median m of $T_n(k)$ and $T_n(k+1)$. If $B_n(k) = 1$ and m is in $\mathcal{F}_{n+1}$, we add m to T_{n+1} . For each value in T_{n+1} we check whether the associated ray is white and add the result to B_{n+1} . When B_{n+1} contains no ones, that is, when none of the rays under consideration turn out to be white, we stop.

We don't need to check whether of a ray is white if its slope is already in T_n and the corresponding value in B_n is 0, because it is automatically 0. Also, for each slope in T_{n+1} , we just need to check the whether the pixel at distance $n+1$ from p is white, since the other part of the ray has already been verified as white in previous steps of the algorithm.

Note that, by Proposition 1, T_n is a subset of $\mathcal{F}_n$ for each step n . So at step n , in the

worst case, $T_n = \mathcal{F}_n$ and thus the algorithm has complexity $O(n^2)$, according to the discussion in Section . The following proposition guarantees that all the mediantants that we have inserted are mediantants between successive terms of a Farey sequence.

Proposition 5

If $B_n(k) = 1$ and $k < \#(T_n)$, then $T_n(k)$ and $T_n(k + 1)$ are successive terms in $\mathcal{F}_n$.

Proof

We will prove it by induction on n . It is obviously true for $n = 1$. Assume for $n - 1$. Assume $B_n(k) = 1$ and $k < \#(T_n)$. Then $T_n(k)$ could have been inserted in two possible ways.

One possibility is that $T_n(k) = T_{n-1}(i)$ for some i , and then $B_{n-1}(i) = 1$. By the induction hypothesis, $T_{n-1}(i)$ and $T_{n-1}(i + 1)$ are successive in $\mathcal{F}_{n-1}$. So $T_n(k + 1)$ is the mediant of $T_{n-1}(i)$ and $T_{n-1}(i + 1)$ if it does belong to F_n , and if it doesn't, it is $T_{n-1}(i + 1)$. In any case, $T_n(k)$ and $T_n(k + 1)$ are successive.

The other possibility is that $T_n(k) = T_{n-1}(j) \oplus T_{n-1}(j + 1)$, with $B_{n-1}(j) = 1$. By the induction hypothesis, $T_{n-1}(j)$ and $T_{n-1}(j + 1)$ are successive in $\mathcal{F}_{n-1}$. So $T_n(k + 1) = T_{n-1}(j + 1)$, hence $T_n(k)$ and $T_n(k + 1)$ are successive. $\square$

Proposition 6

The algorithm iterates through all white line segments with lower extreme p . That is, if $p + (D(r) \cap G_n^I)$ is white, and a/b is the term in $\mathcal{F}_n$ immediately below r , then $a/b \in T_n$.

Proof

We assume, without loss of generality, that $p = (0, 0)$. Let's prove it by induction on n , the length of the line. It is clearly true for $n = 1$ since $T_1 = \mathcal{F}_1$. Assume true for $n - 1$. Let L be a white line of length n . We know by Proposition 3 that $L = D(\frac{a}{b}) \cap G_n^I$ for some $\frac{a}{b} \in F_n$. Let's show that $\frac{a}{b} \in T_n$. We distinguish two cases:

- *If $\frac{a}{b} \in \mathcal{F}_{n-1}$, all pixels in $D(\frac{a}{b}) \cap G_{n-1}^I$ are white. By the induction hypothesis $\frac{a}{b}$ is the j -th term in T_{n-1} , and $S_{n-1}(j) = 1$, so $\frac{a}{b}$ will be inserted in T_n .*
- *If $\frac{a}{b} \notin \mathcal{F}_{n-1}$, then $\frac{a}{b} = \frac{c+c'}{d+d'}$, where $\frac{c}{d}$ and $\frac{c'}{d'}$ are consecutive in $\mathcal{F}_{n-1}$. By the induction hypothesis $\frac{c}{d} = T_{n-1}(j)$ for some j , since by Proposition 3 $D(\frac{c}{d}) \cap G_{n-1}^I = D(\frac{a}{b}) \cap G_{n-1}^I$, and this one is also white, so also $S_{n-1}(j) = 1$. By the previous proposition, if $j < \#(T_{n-1})$, $T_{n-1}(j)$ and $T_{n-1}(j + 1)$ would be successive, so $T_{n-1}(j + 1) = \frac{c'}{d'}$ and we would be done. Suppose, aiming for a contradiction, that $j = \#(T_{n-1})$. If $n = 2$, necessarily $\frac{a}{b} = 1/2$, the only mediant in T_2 , which is not the last one, so we obtain a contradiction. For $n > 2$ we have two cases:*
 - *If $T_{n-1}(j) = T_{n-2}(i) \oplus T_{n-2}(i + 1)$, since $B_{n-2}(i)$ must be 1, $T_{n-2}(i + 1)$ will be inserted after $T_{n-1}(j)$ and $T_{n-1}(j)$ wouldn't be the last one.*
 - *So it must be that $T_{n-1}(j) = T_{n-2}(i)$ for some i .*

Repeating the same argument for $T_{n-2}(i)$ and successively, we eventually arrive to the

case $n = 2$, and we get the desired contradiction. So $j < \#(T_{n-1})$, and we are done.

□

Proposed method for centered segments

We now present an algorithm for computing the longest line segments centered at a given pixel p that consist entirely of white pixels. We give a similar definition for the one given to G_n^I , but in the fifth octant.

Definition 5

Let us denote all bounded points in the fifth octant by

$$G_n^V := \{(x, y) \in \mathbb{Z}^2 : -n \leq x \leq 0, x \leq y \leq 0\}.$$

Proposition 7

If r is any real number such that $\frac{a}{b} < r < \frac{a'}{b'}$, with $\frac{a}{b}$ and $\frac{a'}{b'}$ consecutive in $\mathcal{F}_n$, then

$$D\left(\frac{a'}{b'}\right) \cap G_n^V = D(r) \cap G_n^V.$$

Proposition 8

We have

$$S_n^V := \left\{ D\left(\frac{a}{b}\right) \cap G_n^V : \frac{a}{b} \in F_n \right\} = \{D(r) \cap G_n^V : r \in [0, 1]\}.$$

There is also a bijection, analogous to Proposition 4, between Farey fractions and digital segments in the fifth octant. Based on these results, we can characterize each segment as follows. Let $f_0, \dots, f_k$ be the elements of $\mathcal{F}_n$. Restricted to G_n^I , each interval

$$[f_0, f_1), \dots, [f_{k-1}, f_k), [f_k, f_k]$$

corresponds to a naive digital straight segment in the first octant, while each interval

$$[f_0, f_0], (f_0, f_1], \dots, (f_{k-1}, f_k]$$

corresponds to a naive digital straight segment in G_n^V . Therefore, a naive digital segment centered at $(0, 0)$, restricted to $G_n^I \cup G_n^V$, can be represented by an interval from the sequence

$$[f_0, f_0], (f_0, f_1], [f_1, f_1], \dots, (f_{k-1}, f_k], [f_k, f_k].$$

Algorithm 2 provides a method for calculating the longest white segments centered at a given pixel p . The algorithm proceeds iteratively. At step n , T contains all intervals of slopes that yield white centered segments of length $2n$. Furthermore, the previous discussion guarantees that each interval corresponds exactly to one centered segment.

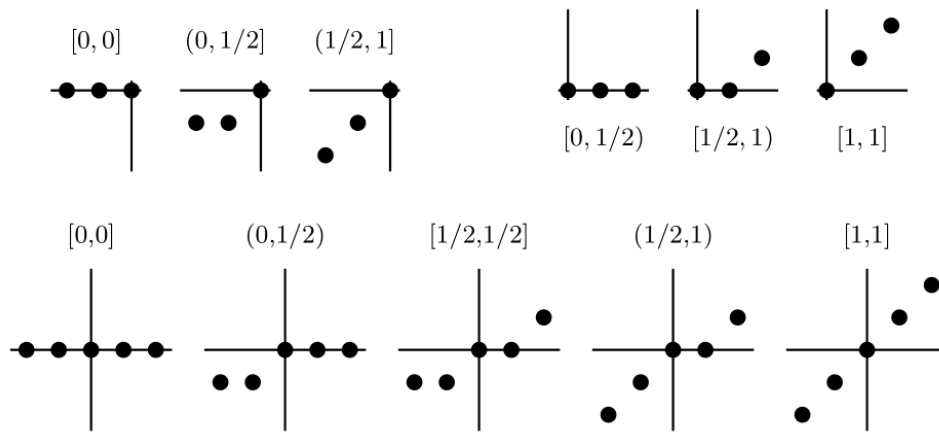


Figure 2: Rays in the fifth and first octant in a bounded grid, in the leftmost and rightmost half of the upper row, respectively, in correspondence with $\mathcal{F}_2$. In the bottom row, centered segments. Each ray or centered segment is marked with the interval of slopes such that if r belongs to that interval, the set $D(r)$ restricted to a suitable domain is equal to the represented set of points.

The vector T' , containing slopes that yield white centered segments of length $2(n+1)$, is built from T as follows. For each interval t in T , if it contains only one slope or if the median m of t_0 and t_1 does not belong to $\mathcal{F}_{n+1}$, it is appended to T' provided it remains white when extended. Otherwise, the interval (t_0, t_1) is subdivided in three parts, (t_0, m) , (m, m) and (m, t_1) , each of which is added to T' provided the corresponding segment is white. If at the end of the step T' is empty, that is, no valid centered segments of length $2(n+1)$ exist, the algorithm stops. Otherwise, T is replaced by T' , n is incremented, and the algorithm continues.

Note that, as in the algorithm for rays, only the farthest pixels from a centered segment need to be checked at each step.

A2: Slope determination and contour processing via Farey ranks

Problem statement

Given a detected straight structure in a digital image (e.g., a segment with endpoints or a local contour portion approximated by a line), the goal is to assign a stable direction descriptor and to compare or cluster directions across segments without relying on floating-point angle binning.

Farey table

A Farey table T_n for $\mathcal{F}_n$ (the Farey sequence of order n) is an $(n+1) \times n$ matrix in which each row a , $0 \leq a \leq n$, represents the numerators and each column b , $1 \leq b \leq n$, represents the denominators (see Figure 3).

The entry $T_n(a, b)$ stores the rank of the fraction $\frac{a}{b}$ in $\mathcal{F}_n$. If a fraction $\frac{a'}{b'}$ reduces to $\frac{a}{b}$, then $T_n(a', b') = T_n(a, b)$ [13]. Once slopes are represented by ranks, directional

		denominator (j)			
		1	2	3	4
numerator (i)	0	1	1	1	1
	1	7	4	3	2
	2		7	5	4
	3			7	6
	4				7

Figure 3: Farey table $T_4 = \{\frac{a}{b} : 0 \leq a \leq b \leq 4\}$, where the 1st column and row represents a and b respectively

comparisons reduce to simple integer subtractions rather than floating-point angle computations.

Properties of the Farey Table

Several properties of the Farey table relevant to the present work are summarized below [4].

1. **Symmetry of ranks.** For any two fractions whose sum is unity, the sum of their ranks equals $f_{\max} + 1$, where $f_{\max}$ denotes the maximum rank in the Farey table.
2. **Row-wise monotonicity.** In a Farey table T_n , the ranks along each row decrease from left to right, except for the first row, in which all entries have rank 1.
3. **Column-wise monotonicity.** The ranks in each column of the Farey table increase monotonically from top to bottom.
4. **Vertical symmetry of rank differences.** For a fixed column m in T_n , the differences between consecutive ranks exhibit vertical symmetry, with corresponding upward and downward patterns.

Farey table helps us to generate the rank of Farey sequence of any order. Also as shown in Figure 4, the plots on variation of numerical value of the fractions with their rank in $\mathcal{F}_n$ of different order. From the plotted results, it can be observed that, except for minor fluctuations, the overall trend of the variation is approximately linear. Furthermore, as the order increases, these fluctuations become less pronounced. Therefore, it can be concluded that the Farey ranks exhibit an almost linear relationship with the numerical values of the corresponding fractions. Therefore, for a fraction $\frac{a}{b}$ with rank k in the Farey sequence $\mathcal{F}_n$, which contains a maximum of $f_{\max}$ fractions, the value of the fraction can be approximated as $\frac{a}{b} \approx \frac{k}{f_{\max}}$. Consequently, two fractions can be compared by examining their respective ranks in the Farey sequence rather than computing and comparing their exact numerical values [4]. In particular, when determining whether two fractions are close to each other within a specified range, floating-point computations can be avoided. Instead, an estimate can be obtained by simply evaluating the difference between the ranks of the given fractions. It should be

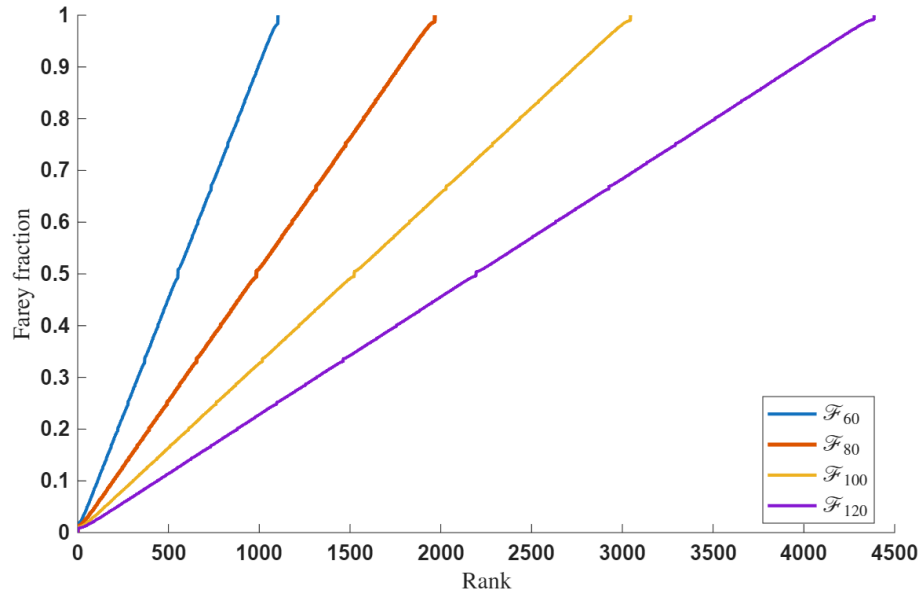


Figure 4: Relationship between Farey ranks and fractions of Farey sequence for the order $\{60, 80, 100, 120\}$, also the reduction of fluctuations when we move from lower to higher orders of Farey sequence.

noted that this rank difference is dependent on the order of the corresponding Farey sequence.

Polygonal Approximations:

In polygonal approximation, a digital contour is represented by a sequence of straight line segments. An exact reconstruction of the contour can be achieved when the number of segments is equal to the number of boundary points [2]. Each segment is associated with a slope, which is converted into a Farey rank using the Farey table.

Adjacent segments are then compared by examining the difference between their corresponding Farey ranks. If the rank difference is small, the two segments have nearly the same direction and can be merged into a single segment. Conversely, if the rank difference is large, a significant change in direction is detected, indicating the presence of a corner or vertex. By adjusting the rank-difference threshold, the contour can be approximated at varying levels of detail.

Slope approximation in Farey fractions: Binary Method

Let us try to search the key fraction $\frac{p}{q}$, $0 \leq p \leq q$ any number in Farey sequence, if the number is not exactly a Farey element so we will approximate its closest number that could contain in Farey elements [13].

Let the gcd of key fraction and order of Farey sequence

$$\gcd\left(\left\lfloor\left(\frac{pn}{q}\right)\right\rfloor, n\right) = g_1 \quad (6)$$

$$\gcd\left(\left\lfloor\left(\frac{pn}{q}\right)\right\rfloor, n\right) = g_2 \quad (7)$$

From Eq. 6 and Eq. 7, assume that respectively

$$\frac{\lfloor(\frac{pn}{q})\rfloor}{n} = \frac{p_1}{q_1} \quad \text{and} \quad \frac{\lceil(\frac{pn}{q})\rceil}{n} = \frac{p_2}{q_2}$$

where $p_1 = \frac{\lfloor(\frac{pn}{q})\rfloor}{g_1}$ and $q_1 = \frac{n}{g_1}$, such that $\gcd(p_1, q_1) = 1$ and also $p_2 = \frac{\lceil(\frac{pn}{q})\rceil}{g_2}$ and $q_2 = \frac{n}{g_2}$, such that $\gcd(p_2, q_2) = 1$.

It is easy to observe that

$$\frac{p_1}{q_1} \leq \frac{p}{q} \leq \frac{p_2}{q_2},$$

since,

$$\lfloor(\frac{pn}{q})\rfloor \leq \lceil(\frac{pn}{q})\rceil.$$

So we have the rank of two terms of Farey sequence which is $T_n(p_1, q_1) = f_1$ and $T_n(p_2, q_2) = f_2$, respectively. The number we want to search is in between these two ranks. As the fractions are in ascending order in Farey sequence so we will use the binary search in $[f_1, f_2]$ to get the fraction closest to $\frac{p}{q}$ in Farey sequence.

In the Figure 5, we compute the difference between each fraction of the Farey sequence having rank between $[f_1, f_2]$ and the key fraction $\frac{p}{q}$. These differences are strictly increasing along the range (left to right). They are negative initially and become positive towards right and passes from the zero-crossing somewhere in between. The closest fraction is at the rank nearest to the zero-crossing.

Traditional binary search locates the closest Farey fraction by repeatedly halving the search interval, ignoring the shape of the difference function. Although reliable, it requires a logarithmic number of steps. If the key fraction is near the upper or lower bound of $\mathcal{F}_n$ then it given results after more iterations. Below is the algorithm and approximated fraction generated by the bisection method.

Slope approximation in Farey fractions: Regula falsi method

The improved Algorithm 5 exploits the near-linearity of the difference-versus-rank curve. By estimating the zero-crossing through interpolation, it typically converges in less iterations[13]. This efficiency is critical in image-processing applications where millions of slope comparisons may be required.

Now we will apply an algorithm using regula falsi method to find that fraction in more efficient way. We get two ranks between which this intersecting point lies. The iteration continues depending on the values of the function at these two points. Due to the linearity of the curve, this intersecting point in all cases is found to be very close to the root of the function. Below is the algorithm and approximated fraction generated by the Regula falsi method.

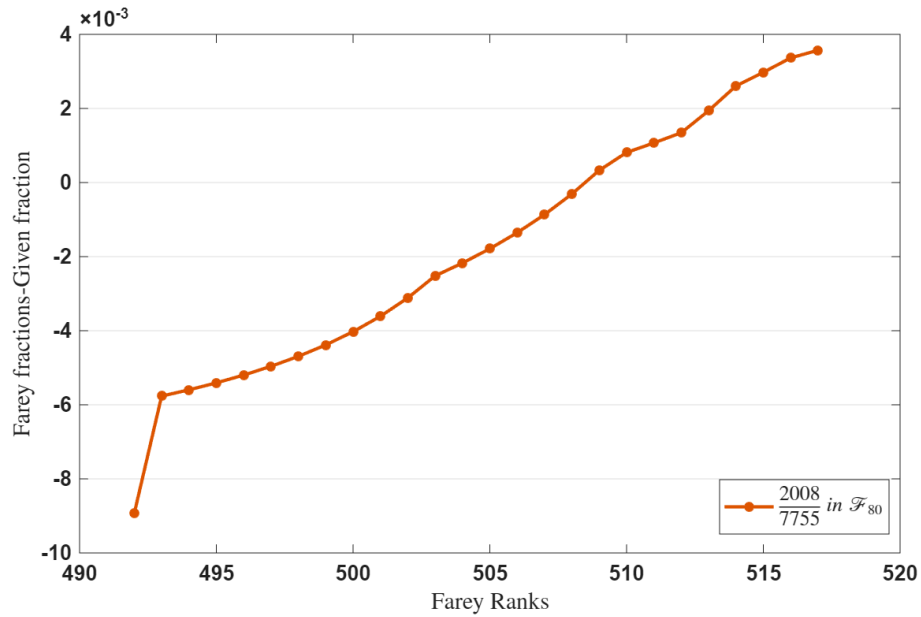


Figure 5: Approximation of the slope $\frac{2008}{7755}$ in $\mathcal{F}_{80}$. We consider the fraction of Farey sequence as the approximating fraction that has less distance from zero crossing

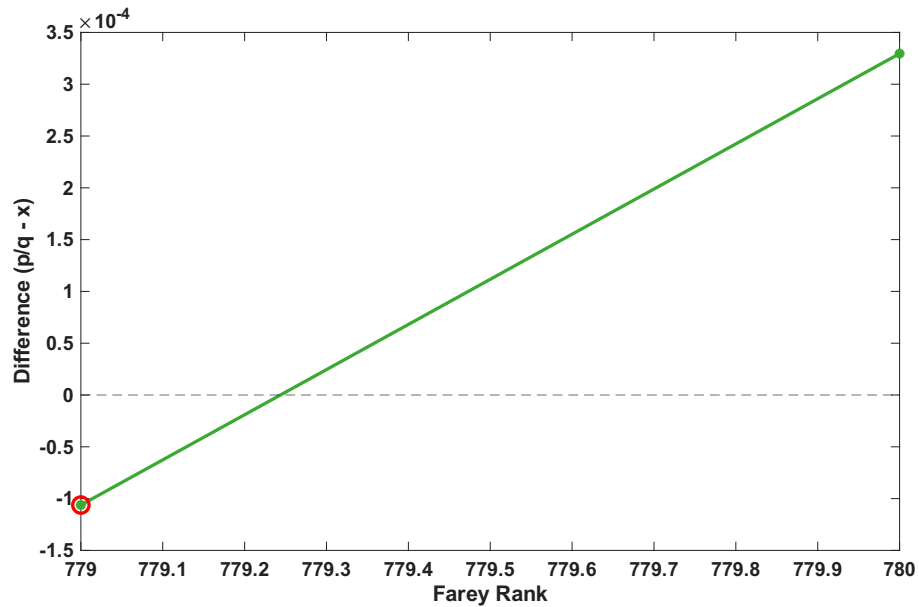


Figure 6: We apply the Bisection method on the binary search technique for finding the closest Farey fraction of $\mathcal{F}_{80}$ from fraction $\frac{2008}{7755}$. Red circle shows the rank of nearest Farey fraction to the slope $\frac{2008}{7755}$ in **6 iterations**. The horizontal dotted line shows the zero-crossing for measuring the distance.

To illustrate the distribution of Farey ranks and the behavior of the Regula Falsi search, Figure 7 presents two complementary visualizations for the Farey sequence.

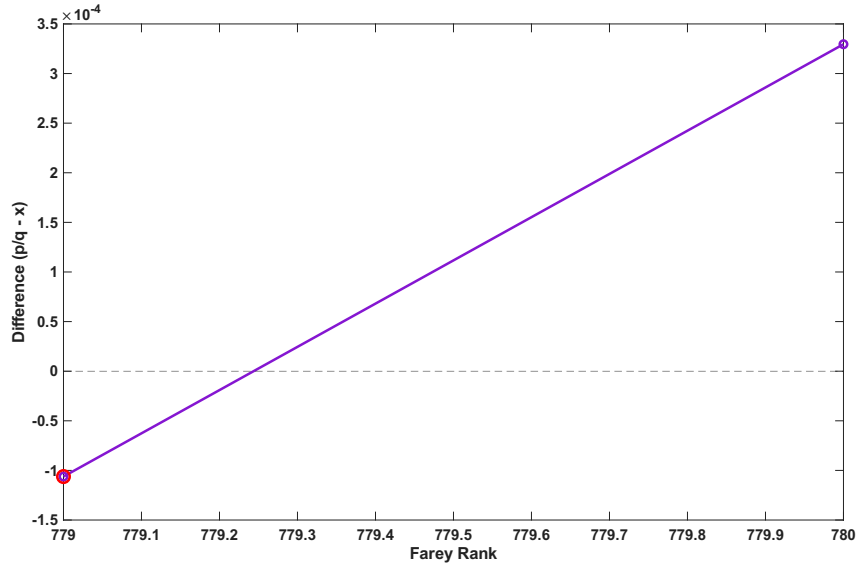


Figure 7: We apply the Bisection method on the binary search technique for finding the closest Farey fraction of $\mathcal{F}_{80}$ from fraction $\frac{2008}{7755}$. Red circle shows the rank of nearest Farey fraction to the slope $\frac{2008}{7755}$ in **3 iterations**. The horizontal dotted line show the zero-crossing for measuring the distance.

Figure 6 and Figure 7 show the experimental results for $\mathcal{F}_{80}$ clearly indicate that while both methods are robust and avoid truncation errors associated with floating-point arithmetic, the regula falsi method offers superior efficiency in terms of iteration count. This advantage is particularly relevant in digital image processing applications, such as boundary detection and shape analysis, where a large number of slope comparisons must be performed rapidly.

Table 1: Comparison of iteration counts for different Farey sequence orders

Slope	Farey order	Binary search iter.	Regula Falsi iter.	Approx. rank
98/180	80	5	2	1072
71/120	100	5	2	1802
375/448	55	4	2	787
2008/7755	99	6	3	779

A3: Farey-discretized rational Hough transform for line detection

Problem statement

Given an image (or an edge map) producing an edge set $E \subset \mathbb{Z}^2$, the goal is to detect dominant straight lines supporting subsets of E under noise and grid/discretization effects (pixel sampling and accumulator binning). We seek a fixed-size output $\mathcal{L}$ consisting

of the top- K line hypotheses (candidate lines parameterized by (θ, ρ) in the classical Hough space or by (a, b, s) in Farey–Hough), while controlling the orientation discretization deterministically. Classical Hough methods discretize θ uniformly, which is convenient but not naturally aligned with the integer lattice. In this application we replace uniform angular binning with a Farey-induced dictionary of *primitive integer normals* and accumulate votes in the integer normal form $ax + by = s$. Optionally, we apply *tolerant voting* along the offset coordinate to improve robustness.

Input edge set

Given an image $I : \Omega \subset \mathbb{Z}^2 \rightarrow [0, 255]$ (grayscale), we extract an edge set $E \subset \Omega$ either by a standard edge detector (e.g., Canny) or, for synthetic/binary images with thick strokes, by a *boundary-only enforcement*. The latter is important because a naive binary threshold can include interior pixels of thick strokes, which creates many spurious accumulator peaks for any Hough-type method.

In boundary-only mode, we compute a discrete gradient magnitude map (e.g., Sobel) $\|\nabla I(x, y)\|$ and define the mask

$$M(x, y) = \mathbf{1} \left\{ \|\nabla I(x, y)\| \geq \text{perc}_p(\{\|\nabla I(u, v)\| : (u, v) \in \Omega\}) \right\}, \quad (8)$$

where $\text{perc}_p(\cdot)$ denotes the p -th percentile (typically $p \in [80, 95]$). We then restrict the edge set to boundary pixels of the thresholded foreground:

$$E = \{(x, y) \in \Omega : I(x, y) > \tau_b \text{ and } M(x, y) = 1\}, \quad (9)$$

with τ_b a binary intensity threshold. This preprocessing step does not change the line parameterization or the voting rule; it enforces that E corresponds to true boundary pixels and reduces spurious votes originating from stroke interiors.

Direction dictionary and parameter space

We build a finite dictionary of *primitive integer normals*, i.e., integer vectors $(a, b) \in \mathbb{Z}^2$ with $\gcd(a, b) = 1$:

$$D_n = \{(a, b) \in \mathbb{Z}^2 : \gcd(a, b) = 1, \text{comp}(a, b) \leq n\}, \quad (10)$$

together with a sign convention to represent each unoriented normal only once (we keep $b > 0$, or $b = 0$ and $a > 0$). The function $\text{comp}(a, b)$ specifies the *complexity bound* that controls the direction budget and can be instantiated in two equivalent ways: (i) a Farey-denominator bound, where slopes $\frac{a}{b}$ are drawn from the Farey sequence $\mathcal{F}_n$ and extended by symmetry to cover $\theta \in [0, \pi)$ (Section); or (ii) a max-norm bound $\text{comp}(a, b) = \max(|a|, |b|)$, i.e., $\max(|a|, |b|) \leq n$. In both cases, the dictionary size grows quadratically, $|D_n| = O(n^2)$, making n an explicit direction-resolution (budget) parameter. Figure 8 visualizes the induced orientation sampling by plotting the normal angles $\theta = \text{atan2}(a, b)$ for two values of n ; the sampling is hierarchical but non-uniform in θ , reflecting the underlying rational/primitive structure rather than uniform angular binning.

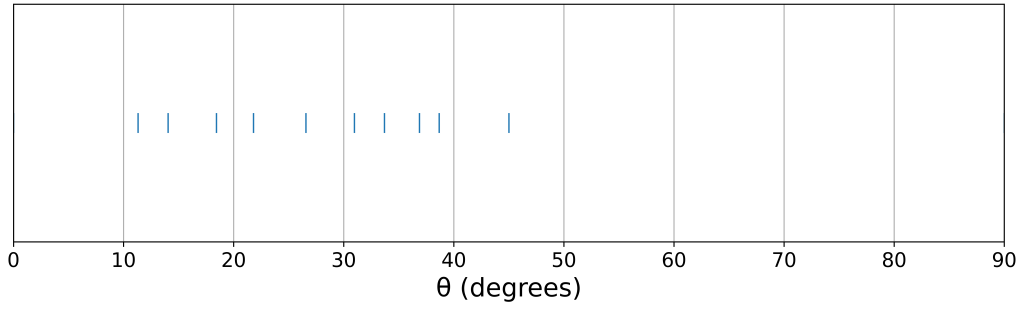
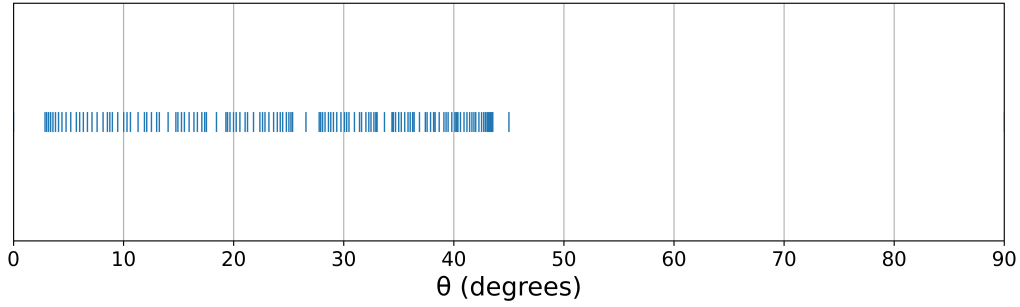
(a) Farey direction sampling of order $n = 5$ (b) Farey direction sampling of order $n = 20$

Figure 8: Farey-inspired direction sampling visualized via normal angles $\theta = \text{atan2}(a, b)$ for two dictionary orders. Increasing n refines the set hierarchically but non-uniformly in θ , reflecting reduced rational (primitive) structure rather than uniform angular bins.

Farey–Hough line model: integer normal form

Instead of parameterizing lines by (θ, ρ) , we use an integer (Diophantine) normal form

$$ax + by = s, \quad (a, b) \in D_n, \gcd(a, b) = 1, s \in \mathbb{Z}, \quad (11)$$

where (a, b) is a primitive integer normal and s is an integer offset. For an edge point $(x, y) \in E \subset \mathbb{Z}^2$ and a fixed normal (a, b) , the induced offset is

$$s_0 = ax + by \in \mathbb{Z}. \quad (12)$$

Since the image domain is finite, only a bounded set of offsets can occur for each direction. Let $\Omega \subset \mathbb{Z}^2$, pixel domain of the image. We define the feasible offset interval

$$[s_{\min}(a, b), s_{\max}(a, b)] = \left[\min_{(x,y) \in \Omega} (ax + by), \max_{(x,y) \in \Omega} (ax + by) \right], \quad (13)$$

and store votes in a 1D accumulator indexed by

$$j = s - s_{\min}(a, b) \in \{0, 1, \dots, s_{\max}(a, b) - s_{\min}(a, b)\}. \quad (14)$$

This indexing yields a compact per-direction histogram in the integer variable s while preserving the exact lattice arithmetic of the model.

Tolerant voting in (a, b, s)

For a fixed tolerance radius $\tau \geq 0$, we define the (tolerant) Farey–Hough accumulator $A_\tau(a, b, s)$ as the vote count associated with the line hypothesis $ax + by = s$, where $(a, b) \in D_n$ and $s \in \mathbb{Z}$ (restricted in practice to the feasible offset range for each direction, Eq. (13) and Eq. (14)). For each edge point $(x, y) \in E$ and each normal $(a, b) \in D_n$, we compute the induced integer offset $s_0 = ax + by$ and accumulate votes using a tolerant rule parameterized by an integer radius $\tau \geq 0$:

$$A_\tau(a, b, s) \ += 1 \quad \text{for all} \quad s \in \{s_0 - \tau, s_0 - \tau + 1, \dots, s_0 + \tau\}. \quad (15)$$

Equivalently, for each fixed direction (a, b) , the per-direction histogram over s is convolved with a box kernel of width $2\tau + 1$. This discrete smoothing increases robustness to grid/discretization effects (offset binning) and to edge localization noise, at the cost of potentially merging nearby parallel lines when τ is too large.

We emphasize that τ is measured in *integer s -bins*. When mapping to the classical normal parameter ρ via $\rho = s/\sqrt{a^2 + b^2}$, one s -bin corresponds to $\Delta\rho = 1/\sqrt{a^2 + b^2}$, so the effective smoothing in ρ becomes direction-dependent.

Peaks, peak splitting, and mapping to (θ, ρ)

Candidate lines correspond to peaks (a, b, s) in the (tolerant) accumulator A_τ . For a fixed direction $(a, b) \in D_n$, we treat $s \mapsto A_\tau(a, b, s)$ as a 1D histogram over feasible offsets, Eq. (13) and Eq. (14). A *peak* is a local maximum of this 1D histogram that exceeds a vote threshold (after optional 1D non-maximum suppression in s ; Section). Two discretization effects are central for interpretation and post-processing: (i) thick edges tend to produce *plateau- or blob-shaped maxima* along s (votes spread across neighboring offsets), and (ii) when the true line orientation is not exactly represented in D_n , votes may spread across nearby rational/primitive approximants, producing *peak splitting across directions*. The latter is illustrated empirically in Figure 16 (Section) and motivates direction-aware suppression and/or geometric verification.

For comparability with the classical normal form $\rho = x \cos \theta + y \sin \theta$, each hypothesis (a, b, s) is mapped to

$$\theta = \text{atan2}(b, a), \quad \rho = \frac{s}{\sqrt{a^2 + b^2}}.$$

This mapping is used only for reporting and evaluation; the accumulator remains integer-indexed in (a, b, s) .

Candidate extraction and non-maximum suppression

For each direction (a, b) we extract candidate offsets by finding local maxima of the 1D histogram $s \mapsto A_\tau(a, b, s)$ above a vote threshold T . To reduce duplicates along the offset axis, we apply 1D non-maximum suppression (NMS) in s with radius r_s , retaining at most one peak within any interval of length $2r_s + 1$. Additionally, to suppress secondary peaks within the same direction, we apply a relative threshold and keep only peaks with votes $v \geq \alpha \max_s A_\tau(a, b, s)$.

Direction-aware suppression (neighbor directions). To mitigate peak splitting and near-duplicate hypotheses across nearby directions, we optionally apply NMS in direction space. We first impose an ordering on D_n (either by Farey slope order when D_n is built from $\mathcal{F}_n$, or simply by increasing normal angle $\theta = \text{atan2}(b, a)$). We then define the neighbors of a direction as the preceding and following directions in this order. In practice, we retain a direction-level hypothesis only if its score (e.g., best peak vote or best inlier count) is locally maximal with respect to its neighbor directions; the effect of this suppression under thickness and noise is shown in Figure 17.

Greedy selection (verification-aware top- K)

Raw accumulator peaks may include near-duplicates or spurious hypotheses, especially for large n or thick/structured edges. Therefore, we optionally select the final K lines using an inlier-based verification step inspired by consensus model fitting [7]. For a candidate line $\ell = (a, b, s)$ we define the point-to-line distance

$$d((x, y), \ell) = \frac{|ax + by - s|}{\sqrt{a^2 + b^2}}, \quad (16)$$

and its inlier set $\mathcal{I}(\ell) = \{(x, y) \in E : d((x, y), \ell) \leq \delta\}$ for a tolerance δ . Given a candidate pool $\mathcal{C}$ extracted from the accumulator, we greedily select up to K hypotheses by repeatedly choosing

$$\ell^* = \arg \max_{\ell \in \mathcal{C}} |\mathcal{I}(\ell)|,$$

adding ℓ^* to $\mathcal{L}$, and removing its inliers from E (i.e., $E \leftarrow E \setminus \mathcal{I}(\ell^*)$). Optionally, we also suppress near-duplicates with respect to already selected lines using small tolerances in (θ, ρ) . This matching-pursuit style procedure suppresses duplicates and prioritizes geometric support over raw accumulator magnitude.

Numerical experiments

This section evaluates the proposed Farey-based framework under controlled and reproducible budgets. We report results separately for the three application settings introduced in Section , and we include sensitivity studies to isolate the effect of the Farey order n .

Implementation and experimental protocol

All methods are implemented in C++ (A1), Matlab (A2), and Python (A3).

A1: Longest digital straight segment detection

Datasets

We tested both Algorithm 1 and Algorithm 2 on square images whose only white pixels were a ray and centered segment, respectively, starting at the central pixel. The algorithms successfully reported the slope and maximum length of the segments.

We generated synthetic binary images, as the one shown in Figure 9, for testing purposes. This particular one, a square image with a width of 40 pixels, was generated by the following rule: paint the pixel (x, y) white if and only if

$$\|(x, y)\| < 0.3\sqrt{2 \cdot 40^2} \text{ and } \|(x - 3, y)\| > 0.2\sqrt{2 \cdot 40^2}.$$

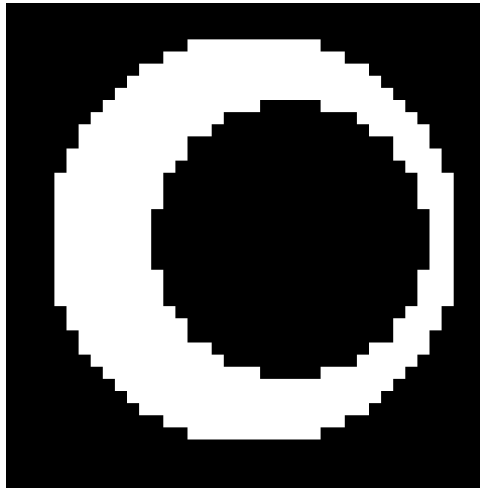


Figure 9: Synthetic image generated for the purpose of testing the line detection algorithms.

Straight segment detection

In Figure 10 we show the output the two algorithms on the image of Figure 9 and two different starting pixels.

A2: Slope and contour-based processing via Farey ranks

Setup: polygonal approximation and turning descriptor

After polygonal approximation, each object contour is represented by a sequence of vertices and edges. At each vertex, two consecutive edges meet, each associated with a direction in the order- n dictionary. We represent these directions by Farey ranks (Section) and define the turning measure at a vertex as the rank difference between consecutive edges (with appropriate wrap-around conventions).

Rotation robustness experiment

To evaluate rotation stability, we rotate each shape by a set of angles and recompute the rank-based turning descriptor. We report the change in the descriptor values before and after rotation. Figure 11 from [4] illustrates a representative example of rotation invariant.

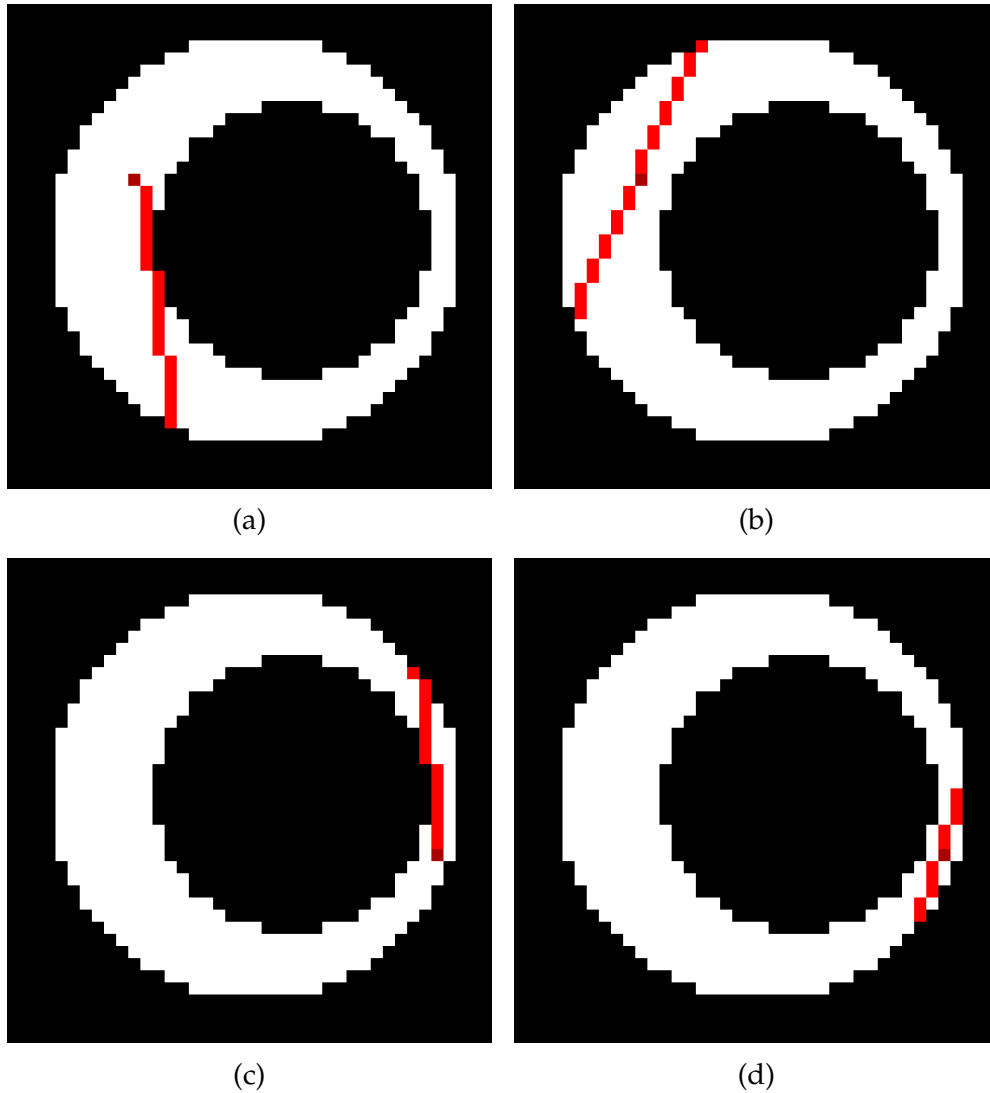
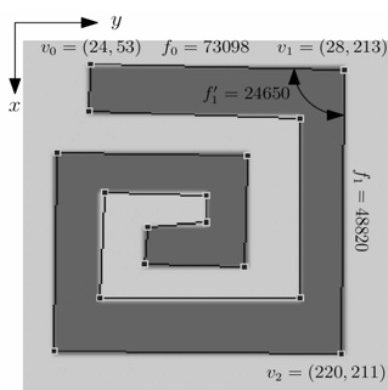
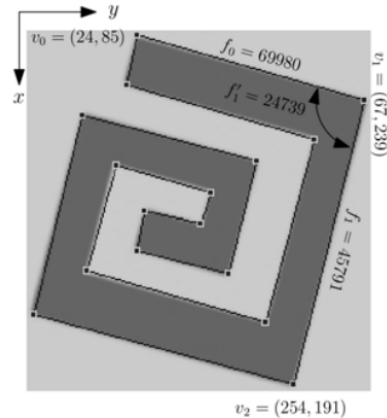


Figure 10: Results for a particular image of the algorithms for finding the longest ray and centered segment through a given pixel, represented in red. The given pixel is painted in dark red. (a) One among the longest rays for pixel (10, 25). The longest rays are in the 7th octant, have length 20 and slopes $\frac{-19}{2}$, $\frac{-9}{1}$, $\frac{-17}{2}$, $\frac{-8}{1}$, $\frac{-15}{2}$ and $\frac{-7}{1}$. (b) Longest centered segment for pixel (10, 25). It has length 22 and slope $\frac{11}{5}$. (c) Longest ray for pixel (35, 15). It is in the third octant, with length 15 and slope $\frac{-15}{2}$. (d) One among the longest centered segments for pixel (35, 15). The longest centered segments have length 10 and slopes $\frac{9}{2}$, $\frac{4}{1}$, $\frac{7}{2}$, $\frac{3}{1}$ (shown) and $\frac{8}{3}$.



(a) Shape feature before rotation



(b) Shape feature after rotation

Figure 11: Rotation robustness of a rank-based turning descriptor derived from Farey directions .

Quantifying angular error

To interpret rank differences in angular units, we optionally normalize by $|\mathcal{F}_n|$ and map to degrees. For example, if a contour vertex yields rank difference f' , we report the proxy angle $\hat{\alpha} = 360^\circ \frac{f'}{|\mathcal{F}_n|}$. We emphasize that this mapping is used only for interpretation; all computations remain rank-based.

A3: Farey-discretized rational Hough transform

We evaluate Farey–Hough against the classical Hough transform [5] under an equal output budget (top- K lines), using identical edge sets whenever possible. Our focus is to characterize: (i) the effect of the direction-resolution parameter n (directional budget), (ii) the robustness parameter τ (tolerant voting), and (iii) the practical role of suppression/verification mechanisms in the presence of peak splitting and thick/noisy edges. Figure 12 shows the input scene used throughout this subsection.

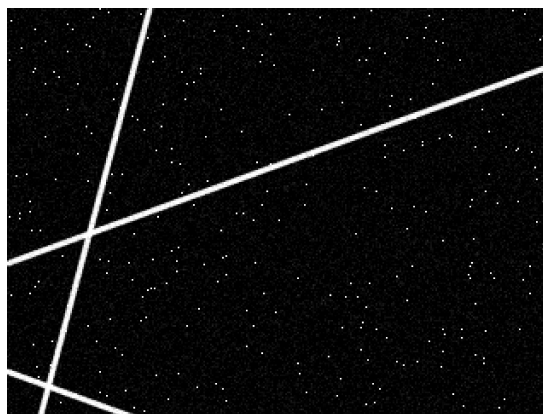


Figure 12: Synthetic input image used in the evaluation (grayscale with additive noise). Subsequent figures show line detections overlaid on the same scene for classical Hough (left, red overlays) and Farey–Hough (right, green overlays).

Table 2: Runtime and edge coverage for Farey–Hough as a function of the direction-resolution parameter n under an equal top- K output budget (top- $K = 3$), compared to classical Hough with a fixed number of θ bins. Coverage is the fraction of Canny edge pixels within distance $\leq \Delta$ of at least one detected line (here $\Delta = 1.5$ pixels).

Method	n / bins	time (ms)	coverage (%)	notes
Classical Hough	#bins=28	1.7	72.26	$(\Delta\theta, \Delta\rho)$ fixed
Farey–Hough	$n = 2$	3.3	9.68	$\tau = 1$
Farey–Hough	$n = 5$	8.0	30.23	$\tau = 1$
Farey–Hough	$n = 10$	18.0	38.48	$\tau = 1$

Directional budget and fair comparison

For Farey–Hough, the directional resolution is controlled by the dictionary order n , which determines the number of primitive normals $|D_n|$ (Section). We enforce an equal output budget by returning the top- K line hypotheses for both methods. For classical Hough, we fix the angular discretization to a representative baseline (here $\#\theta$ bins = 28) and tune the line threshold so that at least K hypotheses are produced before truncation. This protocol isolates the effect of the direction dictionary and tolerant voting from trivial differences in the number of returned lines.

Datasets and benchmark setup

We generate synthetic images with known lines or segments, controlling slope, offset, thickness, additive noise level, and outlier rate. Edge sets are extracted once per image using the Canny detector on the grayscale input, yielding $E \subset \Omega \subset \mathbb{Z}^2$. The same edge set E is then fed to both classical Hough and Farey–Hough to ensure a fair comparison. This setup enables direct measurement of sensitivity to n and τ , and supports diagnostic experiments that isolate accumulator phenomena such as peak splitting (across directions) and blob-shaped maxima (along offsets).

Effect of n (direction resolution)

As n increases, $|D_n|$ grows approximately quadratically, refining the set of available orientations. In our synthetic setting (fixed $\tau = 1$ and top- $K = 3$), increasing n improves the ability to represent oblique lines whose slopes are poorly approximated by coarse dictionaries. The qualitative trend is illustrated in Figure 13, while the quantitative trade-off between runtime and edge coverage is summarized in Table 2. The main trade-off is increased compute and a higher density of near-duplicate hypotheses, which motivates peak suppression and/or verification. To visualize redundancy in the classical baseline, Figure 14 shows a diagnostic run with an intentionally large output budget (large top- K).

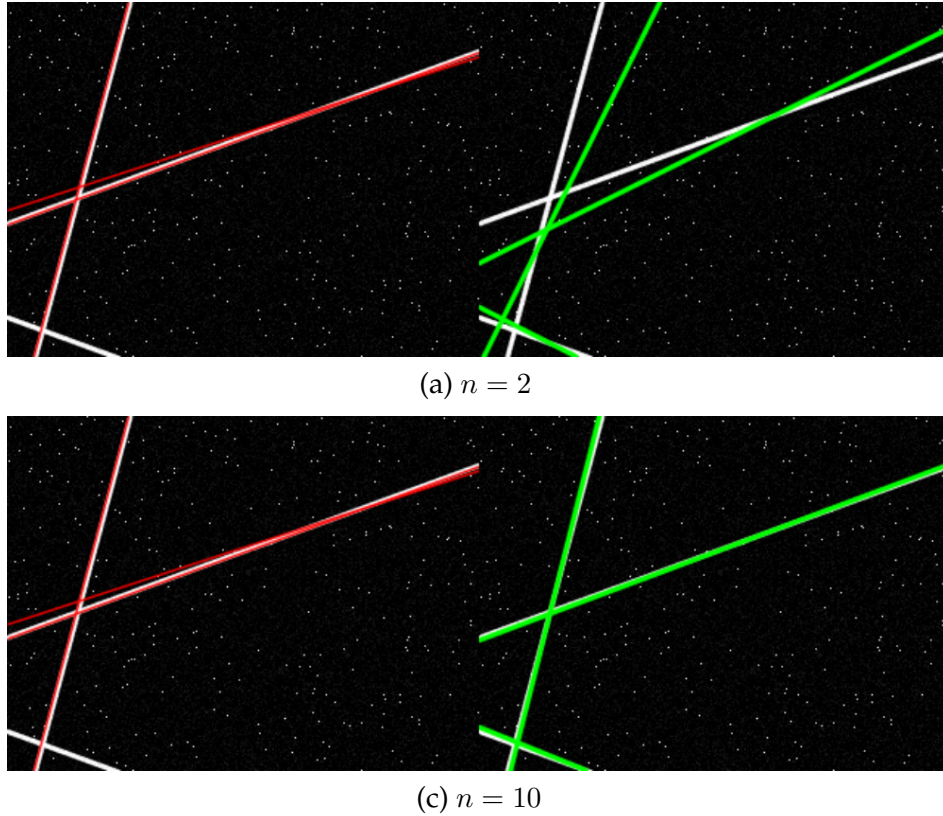


Figure 13: Qualitative effect of the direction-resolution parameter n under an equal top- K output budget (top- $K = 3$). Each panel shows classical Hough (left, red overlays) and Farey-Hough (right, green overlays) computed on the same edge set. Parameters: $\tau = 1$.

Effect of τ (tolerant voting)

We vary the tolerant voting radius τ and quantify robustness to edge localization noise and offset discretization. Figure 15 shows representative values for fixed $n = 10$. Small τ improves separability but is sensitive to discretization, while moderate τ stabilizes peaks and typically increases coverage. When τ is too large, nearby parallel structures may merge because the offset histograms are excessively smoothed. Figure 15 shows representative qualitative outcomes for fixed $n = 10$ and a common top- K budget.

Peak splitting diagnostic (direction space)

To isolate the direction discretization effect, we consider a single oblique line whose true orientation does not coincide with any coarse rational direction. For small n , votes spread across neighboring rational approximants, producing multiple nearby maxima in the direction-score curve (peak splitting). As n increases, the direction set refines and the score concentrates around a closer approximant, reducing splitting, as shown in Figure 16. This motivates NMS in direction space (neighbor-direction suppression) and verification-aware selection when the dictionary is dense.

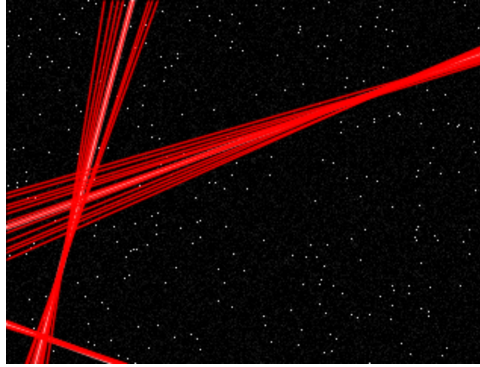


Figure 14: Diagnostic example of redundant detections produced by classical Hough on the synthetic scene when the output budget is intentionally large (large top- K). This motivates verification-aware selection to suppress near-duplicate lines while preserving coverage.

Thickness and noise diagnostic (why direction NMS and verification matter)

Finally, we tested the accumulator under thick edges and sparse noise. Thickness spreads votes across offsets (blob-shaped maxima), while noise creates spurious responses and can elevate weak directions. In this regime, naive top- K selection by raw score may include near-duplicate directions or outlier directions (from noise). Figure 17 shows the edge set and the direction-score curve highlighting the difference between naive top- K selection and neighbor-direction NMS. Figure 18 visualizes the corresponding local accumulator structure, supporting the interpretation that τ controls offset smoothing while NMS/verification controls redundancy and spurious hypotheses.

Cross-application summary

We summarize how the Farey order n acts as a unified parameter across tasks: (A1) n increases as new candidate white rays or centered segments are considered, and (A2 and A3) increasing n consistently improves representational fidelity (segments, slopes, orientations) up to a saturation point, while the computational cost grows predictably with the dictionary size.

Discussion

Why Farey-inspired sampling helps on $\mathbb{Z}^2$.

Discretizing slopes according to Farey sequences allows an exhaustive enumeration of all the possible line segments. Furthermore, primitive integer directions encode slopes that are arithmetically natural on the pixel lattice: simple rational slopes correspond to low-complexity directions, and refinement increases complexity in a structured manner. Compared with uniform θ bins, this yields a discretization whose resolution is linked to rational approximates.

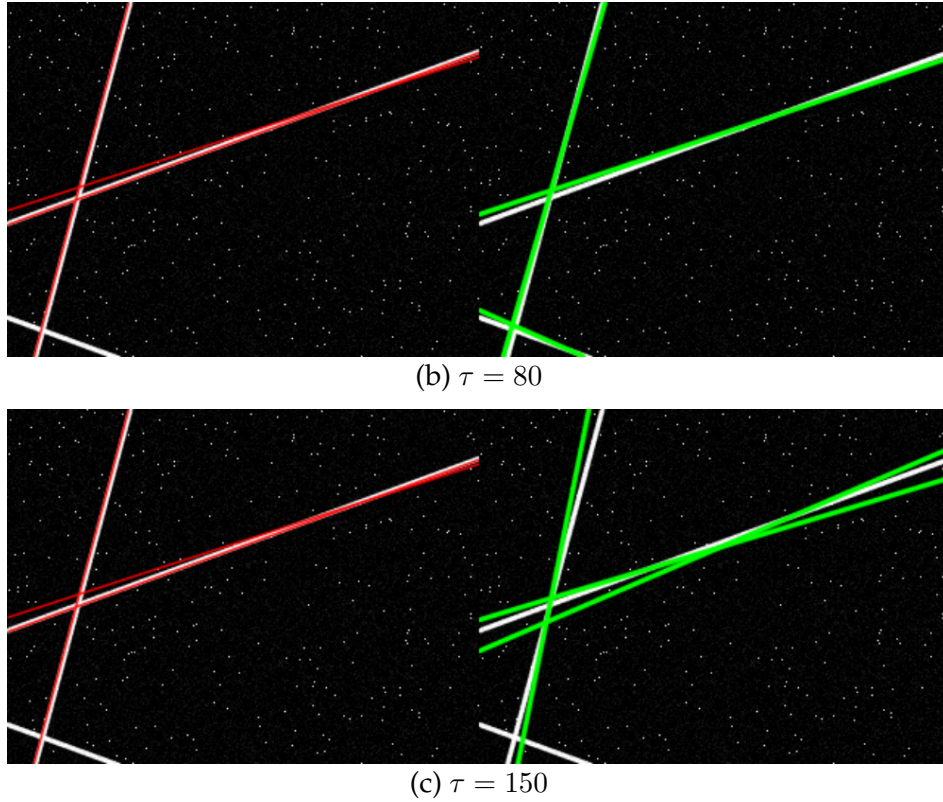


Figure 15: Qualitative effect of tolerant voting τ for fixed direction resolution $n = 10$ under an equal top- K output budget. Each row shows classical Hough (left, red overlays) and Farey-Hough (right, green overlays) computed on the same edge set. As τ increases, offset histograms are increasingly smoothed, improving robustness but potentially merging nearby parallel structures.

Limitations and future work.

In A1, Algorithms 1 and 2 of can be further optimized in time execution: an algorithm could be designed such that Farey sequences are computed in advance. In A2, Farey table puts forward some important issues, such as the problem of compressing the table by removing some columns such that the maximal difference of ranks in the contiguous columns is minimized can be studied further. In A3, the present implementation still inherits typical Hough-family constraints: memory/time grow with the number of directions and with the density of edges, and closely spaced parallel lines may be sensitive to (τ, δ) . Promising directions include progressive refinement in n (coarse-to-fine search), alternative discrete kernels in s (e.g., triangular or robust M-estimator inspired weights), and extensions beyond straight lines (e.g., circles or other algebraic curves) within the same rational dictionary framework.

Conclusion

We presented three applications of Farey sequences for rational discretization in computer vision: detection of the longest digital straight segment in binary images, slope estimation

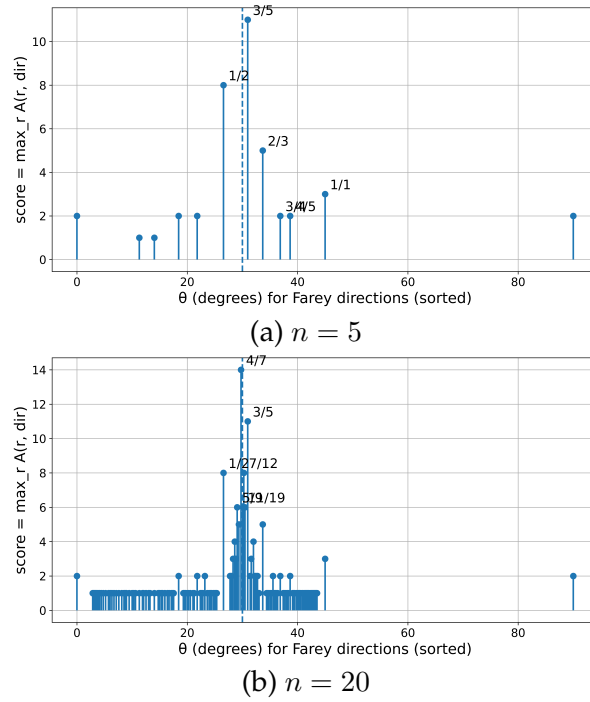


Figure 16: Peak splitting in direction space for a single oblique line: the score $\max_s A_\tau(a, b, s)$ as a function of direction. With small n , the true orientation is approximated by several nearby rationals, yielding multiple competing maxima; increasing n reduces splitting by refining the direction set.

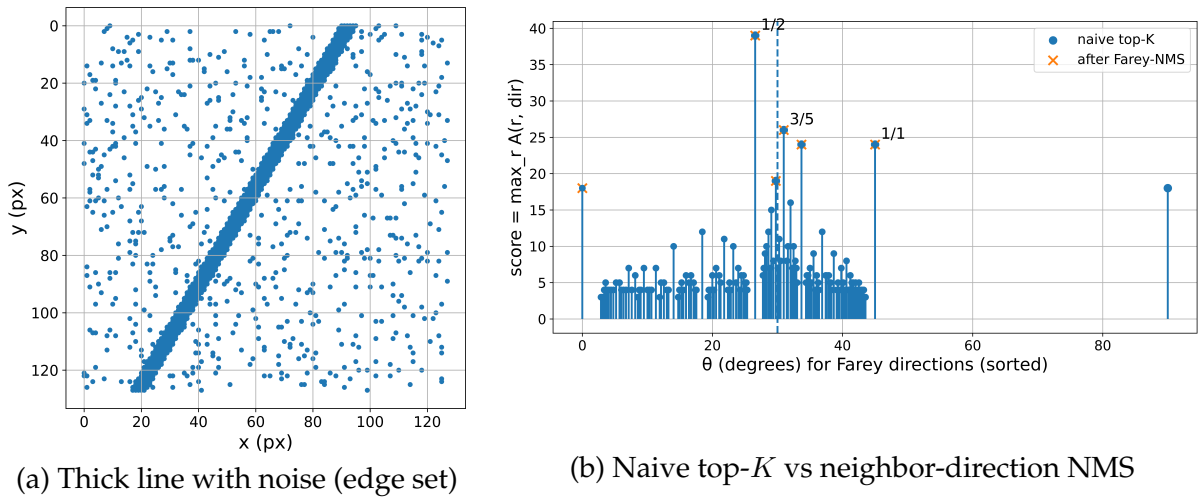
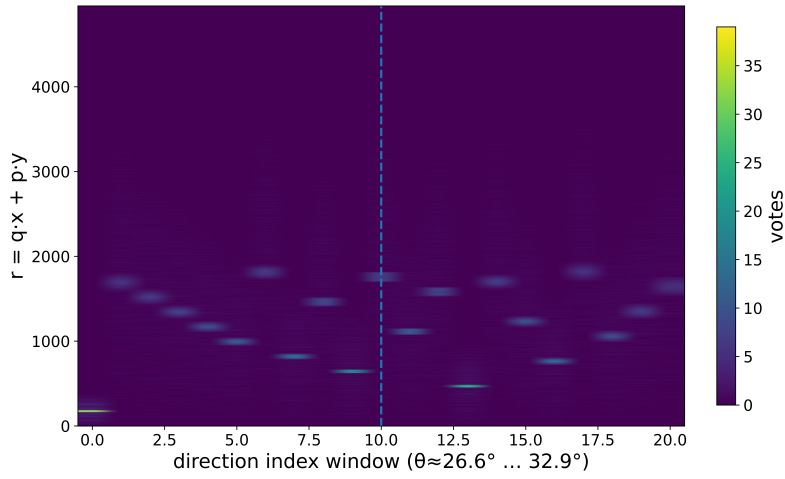
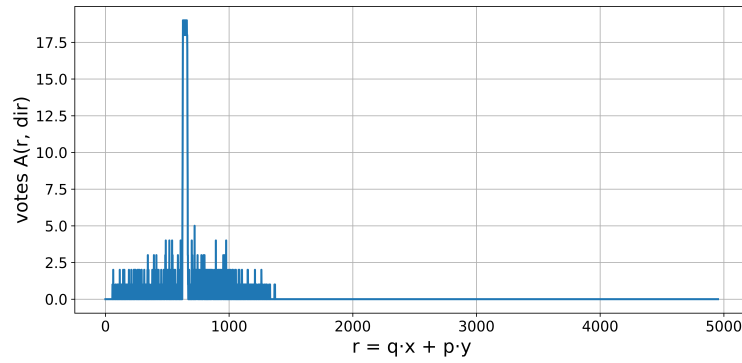


Figure 17: Diagnostic under thickness with noise. (a) Thick oblique line with sparse noise. (b) Direction scores with naive top- K selection versus neighbor-direction NMS, illustrating how direction-aware suppression reduces redundant neighbor directions and improves hypothesis quality before verification/greedy selection.

(a) Zoom of $A_\tau(s, \text{dir})$ around the true direction

(b) Blob-shaped maximum along offset

Figure 18: Local accumulator geometry under thickness with noise. (a) A zoomed view of the accumulator around the true direction reveals a structured neighborhood across directions. (b) For a representative direction, the 1D offset profile shows blob-shaped maxima induced by thickness, motivating tolerant voting τ (controlled smoothing) and verification-aware selection.

for detected straight structures via Farey-based ordering and comparison of rational directions, and a Farey-guided rational Hough transform for line detection in edge maps. In all three cases, the discretization provided by Farey sequences plays a central role for the implementation of the algorithms.

Algorithm 1 Calculation of the longest ray

Require: Binary image I , pixel $p = (x, y)$ **Ensure:** (T, n) , where T is the set of slopes and n the maximal length

```

1: if  $p$  is not white then
2:   return  $(\emptyset, 0)$ 
3: end if
4:  $n \leftarrow 1$ 
5:  $T \leftarrow (0/1, 1/1)$ 
6:  $B \leftarrow$  whether each of the rays for slopes in  $T$  are white
7: if  $B$  contains no ones then
8:   return  $(0/1, 0)$ 
9: end if
10: while true do
11:    $(T', B') \leftarrow (\emptyset, \emptyset)$ 
12:   for all  $k$  in  $\{0, 1, \dots, \#(T) - 1\}$  do
13:     if  $B(k) = 1$  then
14:       add  $(T(k), \text{whether the ray with slope } T(k) \text{ is white})$  to  $(T', B')$ 
15:     else if  $B(k - 1) = 1$  then
16:       add  $(T(k), 0)$  to  $(T', B')$ 
17:     end if
18:     if  $k = \#(T) - 1$  then
19:       break
20:     end if
21:      $m \leftarrow$  median of  $T(k)$  and  $T(k + 1)$ 
22:     if  $B(k) = 1$  and denominator( $m$ )  $\leq n + 1$  then
23:       add  $(m, \text{whether of the ray with slope } m \text{ is white})$  to  $(T', B')$ 
24:     end if
25:   end for
26:   if  $B'$  contains no ones then
27:     break
28:   end if
29:    $n \leftarrow n + 1$ 
30:    $(T, B) \leftarrow (T', B')$ 
31: end while
32: return (slopes in  $T$  corresponding to ones in  $B, n$ )

```

Algorithm 2 Longest centered white segment through a pixel

Require: Binary image I , pixel $p = (x, y)$ **Ensure:** (T, n) , where T is the set of admissible slopes and n the maximal length

```

1: if  $p$  is not white in  $I$  then
2:   return  $(\emptyset, 0)$ 
3: end if
4:  $n \leftarrow 1$ 
5:  $T \leftarrow ([0/1, 0/1], (0/1, 1/1), [1/1, 1/1])$ 
6: remove from  $T$  all  $t$  whose segment of length 2 is not white
7: while true do
8:    $T' \leftarrow \emptyset$ 
9:   for all  $t \in T$  do
10:     $m \leftarrow$  the median of  $t_0$  and  $t_1$ 
11:    if  $t_0 = t_1$  or denominator( $m$ )  $> n + 1$  then
12:      if the segment of length  $2 \cdot (n + 1)$  associated with  $t$  is white then
13:        add  $t$  to  $T'$ 
14:      end if
15:    else
16:      for all  $s \in \{(t_0, m), [m, m], (m, t_1)\}$  do
17:        if the segment of length  $2 \cdot (n + 1)$  associated with  $s$  is white then
18:          add  $s$  to  $T'$ 
19:        end if
20:      end for
21:    end if
22:  end for
23:  if  $T' = \emptyset$  then
24:    break
25:  end if
26:   $T \leftarrow T'$ 
27:   $n \leftarrow n + 1$ 
28: end while
29: return  $(T, n)$ 

```

Algorithm 3 Nearest Farey Fraction Approximation

Require: Target fraction x , Farey order n

Ensure: Approximate fraction p/q

- 1: Generate Farey sequence of order n
- 2: Store fractions in array *Sequence*

▷ Left bound computation

3: $g_1 \leftarrow \gcd(\lfloor xn \rfloor, n)$

4: $p_1 \leftarrow \lfloor xn \rfloor / g_1$

5: $q_1 \leftarrow n / g_1$

6: $l \leftarrow p_1 / q_1$

▷ Right bound computation

7: $g_2 \leftarrow \gcd(\lceil xn \rceil, n)$

8: $p_2 \leftarrow \lceil xn \rceil / g_2$

9: $q_2 \leftarrow n / g_2$

10: $m \leftarrow p_2 / q_2$

11: Find index L such that $Sequence(L) = l$

12: Find index M such that $Sequence(M) = m$

13: **if** $x - l < m - x$ **then**

14: $p/q \leftarrow l$

15: **else**

16: $p/q \leftarrow m$

17: **end if**

18: Compute $N_{\text{seq}} \leftarrow Sequence(L:M) - x$

19: Plot N_{seq} versus indices $L:M$

20: **return** p/q

Algorithm 4 Bisection Method on Farey Sequence

Require: Target fraction x , Farey order n , tolerance ε , maximum iterations $N_{\max}$ **Ensure:** Best rational approximation p/q and Farey rank r

```

1: Generate Farey sequence of order  $n$ 
2: Store fractions in array Sequence
3: Compute reduced left bound  $l$ 
4: Compute reduced right bound  $m$ 
5: Find indices  $r_L$  and  $r_R$  such that  $Sequence(r_L) = l$  and  $Sequence(r_R) = m$ 
6:  $iter \leftarrow 0$ 
7: while  $(r_R - r_L > 1)$  and  $(iter < N_{\max})$  do
8:    $iter \leftarrow iter + 1$ 
9:    $r_{\text{mid}} \leftarrow \lfloor (r_L + r_R)/2 \rfloor$ 
10:   $f_{\text{mid}} \leftarrow Sequence(r_{\text{mid}}) - x$ 
11:  if  $f_{\text{mid}} = 0$  then
12:    break
13:  else
14:    if  $f_{\text{mid}} > 0$  then
15:       $r_R \leftarrow r_{\text{mid}}$ 
16:    else
17:       $r_L \leftarrow r_{\text{mid}}$ 
18:    end if
19:  end if
20: end while
21: if  $|Sequence(r_L) - x| < |Sequence(r_R) - x|$  then
22:    $r \leftarrow r_L$ 
23: else
24:    $r \leftarrow r_R$ 
25: end if
26:  $p/q \leftarrow Sequence(r)$ 
27: return  $p/q, r, iter$ 

```

Algorithm 5 Regula Falsi Search on Farey Sequence

Require: Target fraction x , Farey order n

Ensure: Approximate fraction p/q

```

1: Generate Farey sequence of order  $n$ 
2: Store fractions in array Sequence
3: Compute reduced bounds  $l$  and  $m$ 
4: Find indices  $r_L$  and  $r_R$  in Sequence
5:  $f_L \leftarrow \text{Sequence}(r_L) - x$ 
6:  $f_R \leftarrow \text{Sequence}(r_R) - x$ 
7:  $iter \leftarrow 0$ 
8: while  $r_R - r_L > 1$  &  $iter < N_{\max}$  do
9:    $iter \leftarrow iter + 1$ 
10:   $r \leftarrow r_L - f_L \frac{r_R - r_L}{f_R - f_L}$ 
11:   $r \leftarrow \text{round}(r)$ 
12:   $r \leftarrow \min(\max(r, r_L + 1), r_R - 1)$ 
13:   $f \leftarrow \text{Sequence}(r) - x$ 
14:  if  $f \cdot f_L < 0$  then
15:     $r_R \leftarrow r$ 
16:     $f_R \leftarrow f$ 
17:  else
18:     $r_L \leftarrow r$ 
19:     $f_L \leftarrow f$ 
20:  end if
21: end while
22:  $r \leftarrow \arg \min_{r \in \{r_L, r_R\}} |\text{Sequence}(r) - x|$ 
23: return  $r, \text{Sequence}(r)$ 

```

References

- [1] D. Bailey, Y. Chang and S. Le Moan. Analysing Arbitrary Curves from the Line Hough Transform. *Journal of Imaging*, 6(4):26, 2020.
- [2] P. Bhowmick and B. B. Bhattacharya. Fast polygonal approximation of digital curves. *IEEE TPAMI*, 29(9):1590–1602, 2007.
- [3] R. Decelle, P. Ngo, I. Debled-Rennesson, F. Mothe and F. Longuetaud. Digital Straight Segment Filter for Geometric Description. *DGMM 2021 - International Conference on Discrete Geometry and Mathematical Morphology*, 255–268, 2021.
- [4] S. Das, K. Halder, S. Pratihari and P. Bhowmick. Properties of Farey Sequence and their applications to digital image processing. arXiv:1509.07757, 2015.
- [5] R. O. Duda and P. E. Hart. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communications of the ACM*, 15(1):11–15, 1972.
- [6] T. Ehret and J.-M. Morel. Line Segment Detection: a Review of the 2022 State of the art. *Image Processing On Line*, 14:41–63, 2024.
- [7] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, 1981.
- [8] R. Gioi, J. Jakubowicz, J.-M. Morel and G. Randall. On straight line segment detection. *Journal of Mathematical Imaging and Vision*, 32 (3): 313–347, 2008.
- [9] X. Lin, Y. Zhou, Y. Liu and C. Zhu. A comprehensive review of image line segment detection and description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46 (12): 8074–8093, 2024.
- [10] B. J. Maiseli. Optimization of chamfer masks using Farey sequences and kernel dimensionality. *Scientific Reports*, 12 (1): 7639, 2022.
- [11] S. Pratihari and P. Bhowmick. Vectorization of thick digital lines using Farey sequence. *Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing*: 518–525, 2010.
- [12] S. Pratihari, P. Bhowmick, S. Sural and J. Mukhopadhyay. Skew correction of document images using Farey sequence. *International Journal of Pattern Recognition and Artificial Intelligence*, 27 (7): 1353004, 2013.
- [13] S. Pratihari and P. Bhowmick. On the Farey sequence and its augmentation for image analysis. *International Journal of Applied Mathematics and Computer Science*, 27 (3): 637–658, 2017.
- [14] C. Romanengo, B. Falcidieno and S. Biasotti. Extending the Hough transform to recognize space curves in 3D models. *Computer Aided Geometric Design*, 113: 102377, 2024.

- [15] J. Rothstein y C. Weiman. Parallel and sequential specification of a context sensitive language for straight lines on grids. *Computer Graphics and Image Processing*, 5 (1): 106–124, 1976.
- [16] V. Sukhoy and A. Stoytchev. Formulas and algorithms for Farey sequence length. *Scientific Reports*, 11: 22218, 2021.
- [17] Z. Xu, B.-S. Shin and R. Klette. A statistical method for line segment detection. *Computer Vision and Image Understanding*, 138: 61–73, 2015.