PHYSICS-AWARE SOFT-SENSORS FOR EMBEDDED DIGITAL TWINS

Fabio Marcuzzi[1]

[1]Dipartimento di Matematica “Tullio Levi Civita”,
Universitá di Padova
marcuzzi@math.unipd.it

iNEST DT course, October 14th 2024

A critical example:

ARIANE 5 First Flight Failure - Flight 501, June 4, 1996, European Space Agency, Rocket Launch.mp4

Here applied mathematics matters:

ARIANE 5

Flight 501 Failure

Report by the Inquiry Board

The Chairman of the Board :

Prof. J. L. LIONS

FOREWORD

On 4 June 1996, the maiden flight of the Ariane 5 launcher ended in a failure. Only about 40 seconds after initiation of the flight sequence, at an altitude of about 3700 m, the launcher veered off its flight path, broke up and exploded. Engineers from the Ariane 5 project teams of CNES and Industry immediately started to investigate the failure. Over the following days, the Director General of ESA and the Chairman of CNES set up an independent Inquiry Board and nominated the following members :

- Prof. Jacques-Louis Lions (Chairman) Acad emie des Sciences (France)
- Dr. Lennart L beck (Vice-Chairman) Swedish Space Corporation (Sweden)
- Mr. Jean-Luc Fauquembergue De l egation Ge ne rale pour l'Armement (France)
- Mr. Gilles Kahn Institut National de Recherche en Informatique et en Automatique (INRIA), (France)
- Prof. Dr. Ing. Wolfgang Kubbat Technical University of Darmstadt (Germany)
- Dr. Ing. Stefan Levedag Daimler Benz Aerospace (Germany)

there are **restricted data, real-time requirements and technology constraints**!

A few years later, in 2003, Grieves coins the term "Digital Twin" and in 2010 NASA adopts the digital twin philosophy:

Digital Twin: Manufacturing Excellence through Virtual Factory Replication

This paper introduces the concept of a "Digital Twin" as a virtual representation of what has been produced. Compare a Digital Twin to its engineering design to better understand what was produced versus what was designed, tightening the loop between design and execution.

1.4. Top Technical Challenges

	Technical Challenges in Priority Order
1.	Advanced Mission Systems (TABS 4.5): Adaptive Systems
2.	Integrated System Lifecycle Simulation (TABS 3.2): Full Mission Simulation
3.	Simulation-Based Systems Engineering (TABS 3.3): NASA Digital Twin
4.	Software Modeling (TABS 2.1): Formal analysis and traceability of requirements and design
5.	Integrated Hardware and Software Modeling (TABS 2.2): Advanced Integrated Model V&V
6.	Modeling (TABS 2.4): Cross-scale and inter-regional coupling
7.	Flight Computing (TABS 1.1): System Software for Multi-Core Computing
8.	Integrated Hardware and Software Modeling (TABS 2.2): Complexity Analysis Tools
9.	Flight and Ground Computing (TABS 1.1 and 1.2): Eliminate the Multi-core "Programmability Gap"
10.	Software Modeling (TABS 2.1): Software Verification Algorithms

Embedded Digital Twins[1], that is the virtual representations of physical systems that run in embedded syst are deployed on the edge within the embedded software stack to realize e.g. virtual sensors to enrich available information about physical variables and parameters that cannot be provided by direct physical measurements. These lacking measurements are estimated, at least roughly, by an algorithm that processes the available data, usually called a **soft-sensor**.

Let us call a ***Physics-aware soft-sensor*** the numerical algorithm that performs an indirect measurement by exploiting a **physico-mathematical model** plus a possible **data-driven extension**, used within an **estimation algorithm**.

The complexity of physics-aware soft-sensors depends on:

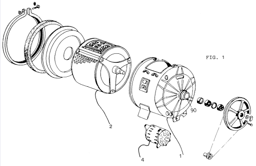
- ◇ model complexity,
- ◇ interactions with the environment,
- ◇ centrality of measured variables in the virtual measurement process.

[1] Herman Van der Auweraer and Dirk Hartmann. The executable digital twin: merging the digital and the physics worlds, Proceedings ISMA2022, International Conference on Noise and Vibration Engineering, Leuven (B), Sept. 12-14, 2022

Physics-aware soft-sensor I: load estimation inside a washing machine through a spinning process

In this example[1], the microprocessor that controls the machine also run a digital twin of the machine mechanical operation, to estimate the weight of the load inserted by the user. The embedded digital twin of each item has been tuned at the end of the production-line (60 pcs/hour), with a least-squares estimation of physical parameters.

Washing Machine



→ **at Production plant: model tuning** of the specific Instance model in the Embedded Digital Twin

→ **at User site: Load Estimation** performed by the Embedded Digital Twin

- ⋄ no model reduction (physics-based lumped model);
- ⋄ online least-squares estimation;
- ⋄ 8-bit microcontroller, tenth seconds for computing.

[1] European Patent 95112646.5-2314 "Improvement in a washing machine with automatic determination of the weight of the washload": https://patents.google.com/patent/EP0704568A1/es

Physics-aware soft-sensor I: load estimation inside a washing machine through a spinning process

European Patent 95112646.5-2314 "Improvement in a washing machine with automatic determination of the weight of the washload" (https://patents.google.com/patent/EP0704568A1/es):

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) **EP 0 704 568 A1**

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
03.04.1996 Bulletin 1996/14

(51) Int. Cl.^{6}: **D06F 39/00**

(21) Application number: **95112646.5**

(22) Date of filing: **11.08.1995**

(84) Designated Contracting States:
DE ES FR GB IT

(30) Priority: **28.09.1994 IT PN940058**

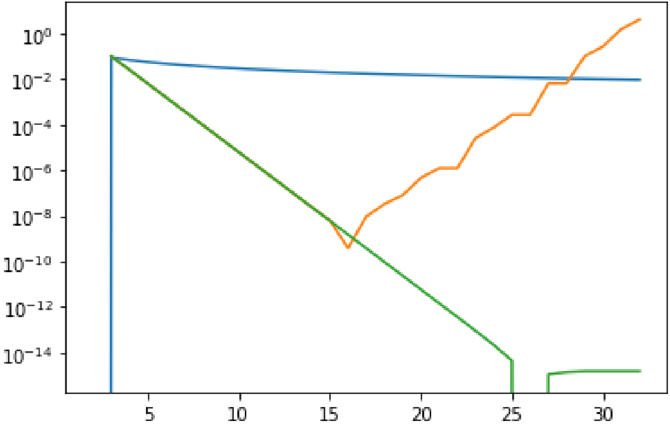
(71) Applicant: **ELECTROLUX ZANUSSI ELETTRODOMESTICI S.p.A.**
I-33170 Pordenone (IT)

(72) Inventors:
- **Marcuzzi, Fabio**
  I-33170 Pordenone (IT)
- **Trangoni, Mario**
  I-33170 Pordenone (IT)
- **Drius, Francesco**
  I-33052 Cervignano, Udine (IT)

$\Rightarrow$ The textitembedded digital twin can be **patented**, iff there is an innovative physical procedure involved.

Physics-aware soft-sensor I: load estimation inside a washing machine through a spinning process

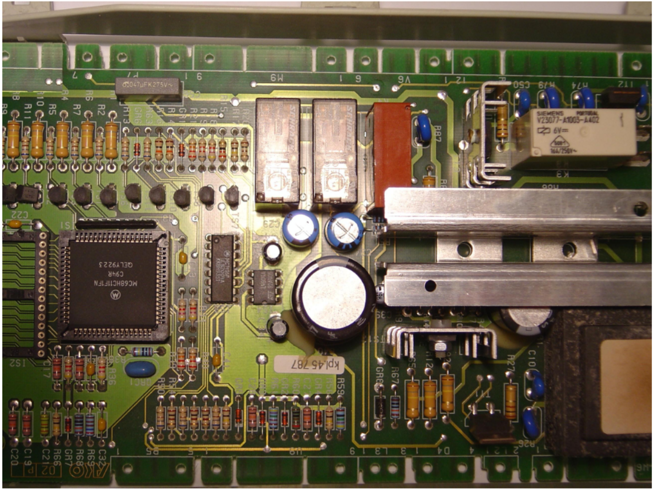
An example of numerical instability:



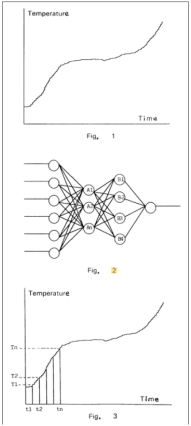
$\Rightarrow$ the **numerical analysis** is necessary!

Physics-aware soft-sensor I: load estimation inside a washing machine through a spinning process

In an *embedded digital twin* the numerical analysis is not sufficient, **the algorithm can suffer from the global behaviour of the system**. Examples: a **cold welding**, an NTC with different thermal inertia, etc.



Physics-aware soft-sensor II: drying-time estimation from temperature measurements



Here an important process variable (air humidity) could not be measured, by the machine embedded controller. Therefore, the physics-based model could not be used and the data-driven model (a **"physical-features-informed neural network"**) tried to understand the physics from the available data and predict the drying time:

$\Rightarrow$ **prediction over Data Driven model Discovery**.

Modeling characteristics of utmost importance:

poor interpretability of the model,

- ⋄ no generalizability,
- ⋄ trustworthiness hard to reach (implicit nonlinear surface response),
- ⋄ high computational efficiency (12-bit arithmetic were sufficient),
- ⋄ no self-adaptation,
- ⋄ 8-bit microcontroller, almost **one minute to forward run the neural network**.

[1] European Patent Application EP95114450A "Improvement in the arrangement used in a clothes drying apparatus to determine the drying time": https://patents.google.com/patent/EP0707107A1/pt

Physics-aware soft-sensor II: drying-time estimation from temperature measurements

European Patent Application EP95114450A "Improvement in the arrangement used in a clothes drying apparatus to determine the drying time": https://patents.google.com/patent/EP0707107A1/pt

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) **EP 0 707 107 A1**

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
17.04.1996 Bulletin 1996/16

(51) Int. Cl.^{6}: **D06F 58/28**

(21) Application number: **95114450.0**

(22) Date of filing: **14.09.1995**

(84) Designated Contracting States:
DE ES FR GB IT

(30) Priority: **11.10.1994 IT PN940061**

(71) Applicant: **ELECTROLUX ZANUSSI ELETTRODOMESTICI S.p.A.**
I-33170 Pordenone (IT)

(72) Inventors:
- **Haberl, Ingo**
  I-33170 Pordenone (IT)
- **Marcuzzi, Fabio**
  I-33170 Pordenone (IT)
- **Drius, Francesco**
  I-33052 Cervignano, Udine (IT)

(74) Representative: **Giugni, Valter et al**
PROPRIA
Protezione Proprietà Industriale S.r.L.
Via Mazzini 13
I-33170 Pordenone (IT)

⇒ **Patent NOT accepted**: the experimental settings was been already patented.

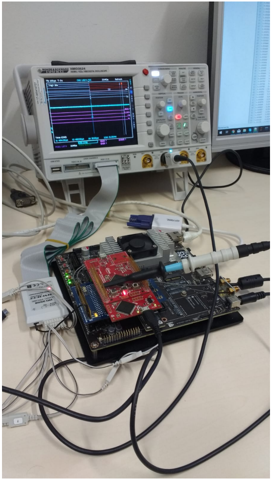
Physics-aware soft-sensor I and II: merging the two approaches?

Scientific machine learning:

- 1990-2000: Lagaris, Likas and Papageorgiou used neural networks to solve partial differential equations;
- 2015: J.S. Hesthaven, G. Rozza, B. Stamm, "Certified Reduced Basis Methods for Parametrized Partial Differential Equations";
- 2019: Raissi, Perdikaris and Karniadakis, "PINNs (Physics-Informed Neural Networks)";
- 2021: S. Wang, H. Wang and P. Perdikaris, "Learning the solution operator of parametric partial differential equations with physics-informed DeepONets";
- 2023: R. Geelen, S. Wright, K. Willcox, "Operator Inference for non-intrusive model reduction with quadratic manifolds";
- 2023: F. Regazzoni, S. Pagani, M. Salvador, L. Dede', A. Quarteroni, "Latent Dynamics Networks: learning the intrinsic dynamics of spatio-temporal processes"
- 2023: I. Gonnella, M. Hess, G. Stabile, G. Rozza, "A two-stage deep learning architecture for model reduction of parametric time-dependent problems";
- 2024: D. Patel, D. Ray, M. R. Abdelmalik, T. J. Hughes, A. A. Oberai, "Variationally Mimetic Operator Networks"

$\Rightarrow$ Let us see how this scenario is evolving in **embedded real-time computing (edge computing)**.

Embedded computing issues for physics-aware soft-sensors



◇ **Numerical accuracy**: from 8-bit microcontrollers, to 32-bit ARM cores (single precision HW f.p.)

◇ **Computing power**: clocks of embedded systems is much lower than PCs.

◇ limited **software libraries for computing**, e.g. numerical linear algebra.

◇ Modelling for physics-aware soft-sensors with differential equations or sophisticated linear algebra implies to **debug** nontrivial numerical models on a real-time system[1].

→ for this reason we developed a tool for **automatic translation** from LaTeX to Python and C, to debug the same code on a PC emulation.

⇒ Moreover, this calls for the implementation of algorithms with a **broad applicability**, like the PAD-NMF is.

[1] Dessole M., Marcuzzi F., Fully iterative ILU preconditioning of the unsteady Navier–Stokes equations for GPGPU (2019) Computers and Mathematics with Applications, 77 (4), pp. 907 - 927

For mechanical/civil structures the digital twin model is a Finite Element model:

... and for embedded digital twins ?!?

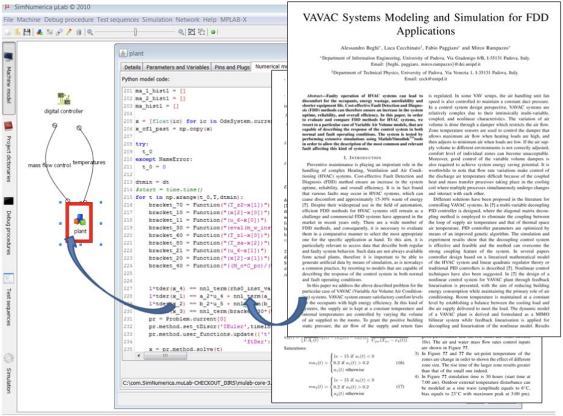


Fig. 5. Automatic translation of the VAVAC model of section V into a python script and its embedding into a co-simulation component.

Access Automation systems: Electric Sliding Gate

The first step for a digital twin is the construction of a mathematical model:

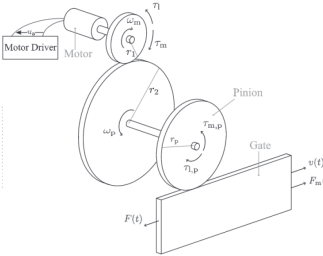
Mathematical Equations (no FOM available):
a DC motor with its drive, reduction gear, pinionrack system, gate:

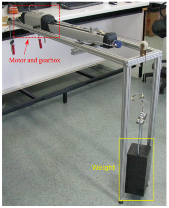
$$J_{eq}\frac{d\omega_m(t)}{dt} + B_{eq}\omega_m(t) = \tau_m(t) - F(t)\frac{N_1 N_g}{\eta_1\eta_g}$$

with

$$J_{eq} = J_m + J_p\frac{N_g^2}{\eta_g} + M\frac{N_1^2 N_g^2}{\eta_1\eta_g}$$

$$B_{eq} = B_m + B_c\frac{N_1^2 N_g^2}{\eta_1\eta_g}$$





Computational inverse problem[1,2]**:**

System Identification and physical parameters estimation (offline).

[1] Marcuzzi, Fabio (2018). Linear estimation of physical parameters with subsampled and delayed data. JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS, vol. 331, p. 11-22

[2] Chiara Faccio, Fabio Marcuzzi (2022). A linear algorithm for the minimal realization problem in physical coordinates with a non-invertible output matrix. LINEAR ALGEBRA AND ITS APPLICATIONS, vol. 644, p. 149-171

Physics-aware soft-sensor III: detection of external forces in a sliding gate

A sensor can detect an effect, but a digital twin, thank to the mathematical model of the system, can describe the **dependencies** and connect **causes and consequences** of the observed phenomena.

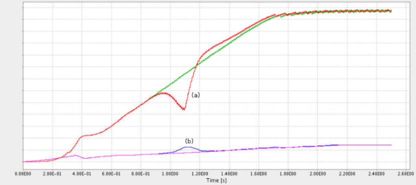
Access Automation systems: Electric Sliding Gate[1]

Delayed opening: the opening of the gate took more time than expected, why?

unexpected friction on the transmission

irregular generation of the motor torque

Temporary exogenous friction: the digital twin compares the predicted and the measured behaviours (see Figure from[1]) and, using the mathematical model, can estimate the exogenous cause of the discrepancy. [2]



green: predicted speed, red: measured speed, magenta: predicted motor current, blue: measured motor current.

[1] A. Beghi, F. Marcuzzi, P. Martin, F. Tinazzi and M. Zigliotto, Virtual prototyping of embedded control software in mechatronic systems: A case study. *Mechatronics* (2017) **43**:99–111.

μLab

The mathematical model for a digital twin can be used preliminarily to implement and verify the control software in a virtual prototype and then to transfer the control software directly into the physical system:

barriera che si muove con firmware sviluppato con muLab.mp4

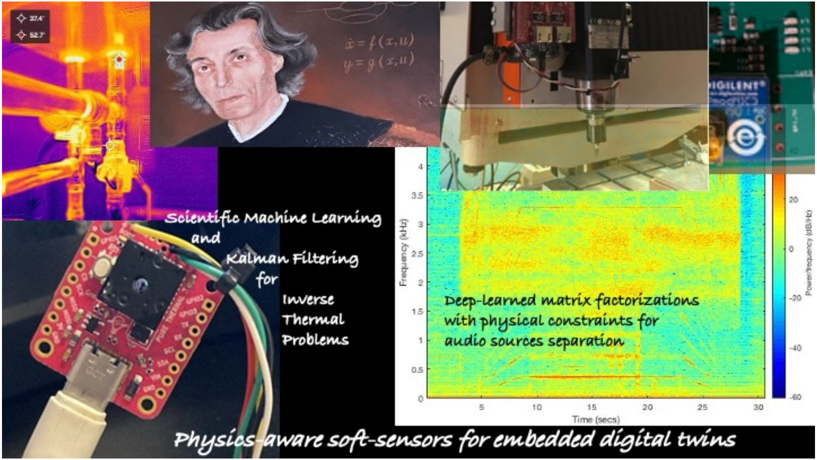
eCfL

and its present version:

eCfL Digital Twin example.mp4

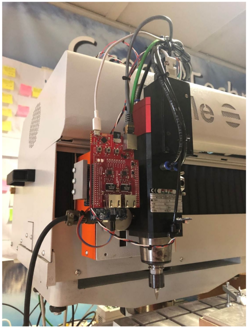
Recent, multi-disciplinary convergence in embedded computing

New sensors and 32-bit CPUs with HW FP units **empower new algorithms** based on systems theory, scientific machine learning, ...



Physics-aware soft-sensor III: hits detection from audio source separation

In experimental vibration analysis, using a **microphone** instead of the usual accelerometer, **the goal is to separate the acoustic created by the process to be monitored** $\{s_t\}$**, from that generated by the environment** $\{n_t\}$.

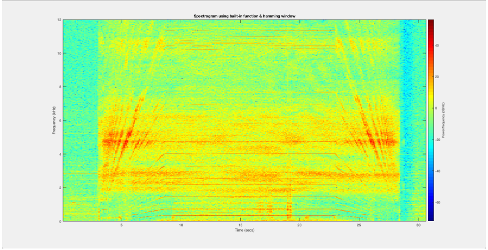


The quantity we measure is a mixture $\{x_t\}$, given by the sum of the two sources:

$$x_t = s_t + n_t$$

Taking the elementwise squared norm of the STFT matrices for the above signals, yields an additive decomposition into matrix spectrograms:

$$X \approx S + N$$



Deep-NMF[I]: Introduction

Let us consider a regularized NMF problem in the form

$$
\begin{aligned}
\text{minimize} \quad & D_1(X, WH) + \square \|H\|_1 \\
\text{subject to} \quad & W \in \mathcal{M}_{M\times R}(\mathbb{R}^+) \quad , \quad H \in \mathcal{M}_{R\times N}(\mathbb{R}^+)
\end{aligned}
$$

which is tackled by alternate optimization of the factors

$$
H^{(k+1)} = f(X; W^{(k)}, H^{(k)}) \quad , \quad W^{(k+1)} = g(X; W^{(k)}, H^{(k+1)})
$$

[I] Hershey J. R., Le Roux J., Weninger F. - *Deep Unfolding: Model-Based Inspiration of Novel Deep Architectures*, (2014)

Deep-NMF[I]: Introduction

Let us consider a regularized NMF problem in the form

$$\begin{aligned} \text{minimize} \quad & D_1(X, WH) + \square \|H\|_1 \\ \text{subject to} \quad & W \in \mathcal{M}_{M\times R}(\mathbb{R}^+) \quad , \quad H \in \mathcal{M}_{R\times N}(\mathbb{R}^+) \end{aligned}$$

which is tackled by alternate optimization of the factors

$$H^{(k+1)} = f(X; W^{(k)}, H^{(k)}) \quad , \quad W^{(k+1)} = g(X; W^{(k)}, H^{(k+1)})$$

By **interpreting the iterative update scheme as a neural network**, where $H^{(k+1)}$ is the output of the k-th layer given the input $H^{(k)}$ and activation function f, Deep-NMF tries to address the convergence issue typical of NMF by *untying* the bases across layers.

$$\begin{tikzcd}
H^{(0)} \arrow[r, "W^{(0)}"] & H^{(1)} \arrow[r, "W^{(1)}"] & \cdots \arrow[r, "W^{(K-2)}"] & H^{(K-1)} \arrow[r, "W^{(K-1)}"] & H^{(K)} \arrow[r, dashed] & \mathcal{E}(W^{(K)}, H^{(K)}) \\
X \arrow[r] & {} \arrow[u] \arrow[r] & \cdots \arrow[r] & {} \arrow[u] \arrow[r] & {} \arrow[u] \arrow[r, dashed] & {} \arrow[u, dashed]
\end{tikzcd}$$

[I] Hershey J. R., Le Roux J., Weninger F. - *Deep Unfolding: Model-Based Inspiration of Novel Deep Architectures*, (2014)

Deep-NMF[I]: Introduction

Let us consider a regularized NMF problem in the form

$$\begin{aligned} &\text{minimize} && D_1(X, WH) + \square \|H\|_1 \\ &\text{subject to} && W \in \mathcal{M}_{M\times R}(\mathbb{R}^+) \quad , \quad H \in \mathcal{M}_{R\times N}(\mathbb{R}^+) \end{aligned}$$

which is tackled by alternate optimization of the factors

$$H^{(k+1)} = f(X; W^{(k)}, H^{(k)}) \quad , \quad W^{(k+1)} = g(X; W^{(k)}, H^{(k+1)})$$

By **interpreting the iterative update scheme as a neural network**, where $H^{(k+1)}$ is the output of the k-th layer given the input $H^{(k)}$ and activation function f, Deep-NMF tries to address the convergence issue typical of NMF by *untying* the bases across layers.

$$\begin{tikzcd} H^{(0)} \arrow[r, "W^{(0)}"] & H^{(1)} \arrow[r, "W^{(1)}"] & \cdots \arrow[r, "W^{(K-2)}"] & H^{(K-1)} \arrow[r, "W^{(K-1)}"] & H^{(K)} \arrow[r, dashed] & \mathcal{E}(W^{(K)}, H^{(K)}) \\ X \arrow[r] & {} \arrow[u] \arrow[r] & \cdots \arrow[r] & {} \arrow[u] \arrow[r] & {} \arrow[u] \arrow[r, dashed] & {} \arrow[u, dashed] \end{tikzcd}$$

Denoting S the clean source spectrogram we want to separate and reconstruct from the mixture, $W_S^{(K)}$ the subdictionary allocated to describe clean frequencies during the training phase and $H_S^{(K)}$ the corresponding rows in the coefficient's matrix, a possible choice for the loss-function $\mathcal{E}$ is

$$\mathcal{E} = \|S - F \circ X\|_2^2 \quad , \quad F = \frac{W_S^{(K)} H_S^{(K)}}{W^{(K)} H^{(K)}}$$

[I] Hershey J. R., Le Roux J., Weninger F. - *Deep Unfolding: Model-Based Inspiration of Novel Deep Architectures*, (2014)

Deep-NMF: A peek under the hood

The backpropagation algorithm is non-conventional, since the weights $W^{(k)}$ **must remain nonnegative**...

Deep-NMF: A peek under the hood

The backpropagation algorithm is non-conventional, since the weights $W^{(k)}$ **must remain nonnegative**...
...so we split the gradients into positive/negative part and recursively compute $\left[\nabla_{H^{(k)}}\mathcal{E}\right]_{\pm}$...

$$
\begin{aligned}
\left[\nabla_{H^{(k)}}\mathcal{E}\right]_{\pm} = & \frac{{W^{(k)}}^{\top}\dfrac{X}{W^{(k)}H^{(k)}}}{{W^{(k)}}^{\top}\mathbb{1}_{N\times M}+\square}\circ\left[\nabla_{H^{(k+1)}}\mathcal{E}\right]_{\pm} \\
& + {W^{(k)}}^{\top}\left(\frac{X}{\left(W^{(k)}H^{(k)}\right)^{\circ 2}}\circ\left(W^{(k)}\left(\frac{H^{(k)}}{{W^{(k)}}^{\top}\mathbb{1}_{N\times M}+\square}\circ\left[\nabla_{H^{(k+1)}}\mathcal{E}\right]_{\mp}\right)\right)\right)
\end{aligned}
$$

Deep-NMF: A peek under the hood

The backpropagation algorithm is non-conventional, since the weights $W^{(k)}$ **must remain nonnegative**...
...so we split the gradients into positive/negative part and recursively compute $[\nabla_{H^{(k)}}\mathcal{E}]_{\pm}$...

$$
\begin{aligned}
[\nabla_{H^{(k)}}\mathcal{E}]_{\pm} \;=\; & \frac{W^{(k)\top}\dfrac{X}{W^{(k)}H^{(k)}}}{W^{(k)\top}\mathbb{1}_{N\times M}+\square}\circ[\nabla_{H^{(k+1)}}\mathcal{E}]_{\pm} \\
& + W^{(k)\top}\left(\frac{X}{\left(W^{(k)}H^{(k)}\right)^{\circ 2}}\circ\left(W^{(k)}\left(\frac{H^{(k)}}{W^{(k)\top}\mathbb{1}_{N\times M}+\square}\circ[\nabla_{H^{(k+1)}}\mathcal{E}]_{\mp}\right)\right)\right)
\end{aligned}
$$

...in order to obtain $[\nabla_{W^{(k)}}\mathcal{E}]_{\pm}$...

$$
\begin{aligned}
[\nabla_{W^{(k)}}\mathcal{E}]_{\pm} \;=\; & \frac{X}{W^{(k)}H^{(k)}}\left(\frac{H^{(k)}}{W^{(k)\top}\mathbb{1}_{N\times M}+\square}\circ[\nabla_{H^{(k+1)}}\mathcal{E}]_{\pm}\right)^{\top} \\
& + \left(\frac{X}{\left(W^{(k)}H^{(k)}\right)^{\circ 2}}\circ\left(W^{(k)}\left(\frac{H^{(k)}}{W^{(k)\top}\mathbb{1}_{N\times M}+\square}\circ[\nabla_{H^{(k+1)}}\mathcal{E}]_{\mp}\right)\right)\right)H^{(k)\top} \\
& + \mathbb{1}_{N\times M}\left(\frac{H^{(k)}}{\left(W^{(k)\top}\mathbb{1}_{N\times M}+\square\right)^{\circ 2}}\circ\left(W^{(k)\top}\frac{X}{W^{(k)}H^{(k)}}\right)\circ[\nabla_{H^{(k+1)}}\mathcal{E}]_{\mp}\right)^{\top}
\end{aligned}
$$

Deep-NMF: A peek under the hood

The backpropagation algorithm is non-conventional, since the weights $W^{(k)}$ **must remain nonnegative**...
...so we split the gradients into positive/negative part and recursively compute $[\nabla_{H^{(k)}}\mathcal{E}]_{\pm}$...

$$
\begin{aligned}
[\nabla_{H^{(k)}}\mathcal{E}]_{\pm} = & \frac{W^{(k)\top}\dfrac{X}{W^{(k)}H^{(k)}}}{W^{(k)\top}\mathbb{1}_{N\times M}+\square} \circ [\nabla_{H^{(k+1)}}\mathcal{E}]_{\pm} \\
& + W^{(k)\top}\left(\frac{X}{(W^{(k)}H^{(k)})^{\circ 2}} \circ \left(W^{(k)}\left(\frac{H^{(k)}}{W^{(k)\top}\mathbb{1}_{N\times M}+\square} \circ [\nabla_{H^{(k+1)}}\mathcal{E}]_{\mp}\right)\right)\right)
\end{aligned}
$$

...in order to obtain $[\nabla_{W^{(k)}}\mathcal{E}]_{\pm}$...

$$
\begin{aligned}
[\nabla_{W^{(k)}}\mathcal{E}]_{\pm} = & \frac{X}{W^{(k)}H^{(k)}}\left(\frac{H^{(k)}}{W^{(k)\top}\mathbb{1}_{N\times M}+\square} \circ [\nabla_{H^{(k+1)}}\mathcal{E}]_{\pm}\right)^{\top} \\
& + \left(\frac{X}{(W^{(k)}H^{(k)})^{\circ 2}} \circ \left(W^{(k)}\left(\frac{H^{(k)}}{W^{(k)\top}\mathbb{1}_{N\times M}+\square} \circ [\nabla_{H^{(k+1)}}\mathcal{E}]_{\mp}\right)\right)\right) H^{(k)\top} \\
& + \mathbb{1}_{N\times M}\left(\frac{H^{(k)}}{\left(W^{(k)\top}\mathbb{1}_{N\times M}+\square\right)^{\circ 2}} \circ \left(W^{(k)\top}\frac{X}{W^{(k)}H^{(k)}}\right) \circ [\nabla_{H^{(k+1)}}\mathcal{E}]_{\mp}\right)^{\top}
\end{aligned}
$$

...and finally update the weights.

$$
W^{(k)} \Leftarrow W^{(k)} \circ \frac{[\nabla_{W^{(k)}}\mathcal{E}]_{-}}{[\nabla_{W^{(k)}}\mathcal{E}]_{+}}
$$

Physics-aware Deep-NMF[2]

The Deep-NMF algorithm and architecture was specifically designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean spectrograms S have strong **intrinsic structure**.

We proposed several **physics-aware enhancements**:

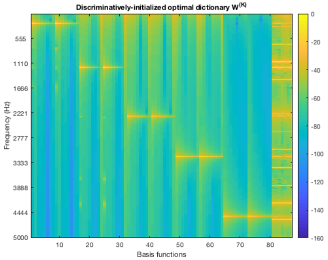
[2] Erik Chinellato and Fabio Marcuzzi. Hits detection in audio mixtures by means of a physics-aware Deep-NMF algorithm, in review, 2024.

Physics-aware Deep-NMF[2]

The Deep-NMF algorithm and architecture was specifically designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean spectrograms S have strong **intrinsic structure**.

We proposed several **physics-aware enhancements**:

- Creation of an **optimal discriminative dictionary** $W^{(K)} = \hat{W}$ (not modified by back-propagation) which enables the recognition of physically-characterized hit responses;

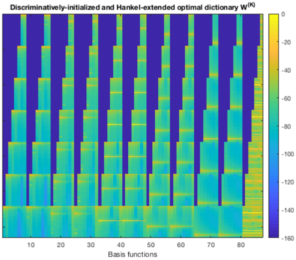


[2] Erik Chinellato and Fabio Marcuzzi. Hits detection in audio mixtures by means of a physics-aware Deep-NMF algorithm, in review, 2024.

The Deep-NMF algorithm and architecture was specifically designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean spectrograms S have strong **intrinsic structure**.

We proposed several **physics-aware enhancements**:

- Creation of an **optimal discriminative dictionary** $W^{(K)} = \hat{W}$ (not modified by back-propagation) which enables the recognition of physically-characterized hit responses;
- Embedding of the time-correlation between consecutive frames in the spectrograms by employing **block-Hankel** weights $W^{(k)}$;



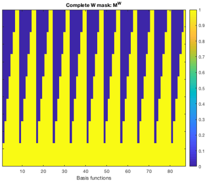
[2] Erik Chinellato and Fabio Marcuzzi. Hits detection in audio mixtures by means of a physics-aware Deep-NMF algorithm, in review, 2024.

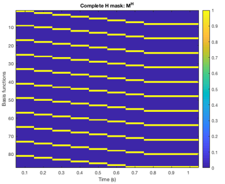
Physics-aware Deep-NMF[2]

The Deep-NMF algorithm and architecture was specifically designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean spectrograms S have strong **intrinsic structure**.

We proposed several **physics-aware enhancements**:

- Creation of an **optimal discriminative dictionary** $W^{(K)} = \hat{W}$ (not modified by back-propagation) which enables the recognition of physically-characterized hit responses;
- Embedding of the time-correlation between consecutive frames in the spectrograms by employing **block-Hankel** weights $W^{(k)}$;
- Preservation of the block-Hankel structure by **projection**, thus modifying both the forward- and back-propagation;





[2] Erik Chinellato and Fabio Marcuzzi. Hits detection in audio mixtures by means of a physics-aware Deep-NMF algorithm, in review, 2024.

Physics-aware Deep-NMF[2]

The Deep-NMF algorithm and architecture was specifically designed for speech enhancement, which effectively makes it ill-suited to deal with datasets stemming from *physico-mathematical models*, where the clean spectrograms S have strong **intrinsic structure**.

We proposed several **physics-aware enhancements**:

- Creation of an **optimal discriminative dictionary** $W^{(K)} = \hat{W}$ (not modified by back-propagation) which enables the recognition of physically-characterized hit responses;
- Embedding of the time-correlation between consecutive frames in the spectrograms by employing **block-Hankel** weights $W^{(k)}$;
- Preservation of the block-Hankel structure by **projection**, thus modifying both the forward- and back-propagation;
- Construction of a **better-suited loss function** for the training process, enforcing a more accurate reconstruction of the clean spectrogram.

Penalty hyperparameter

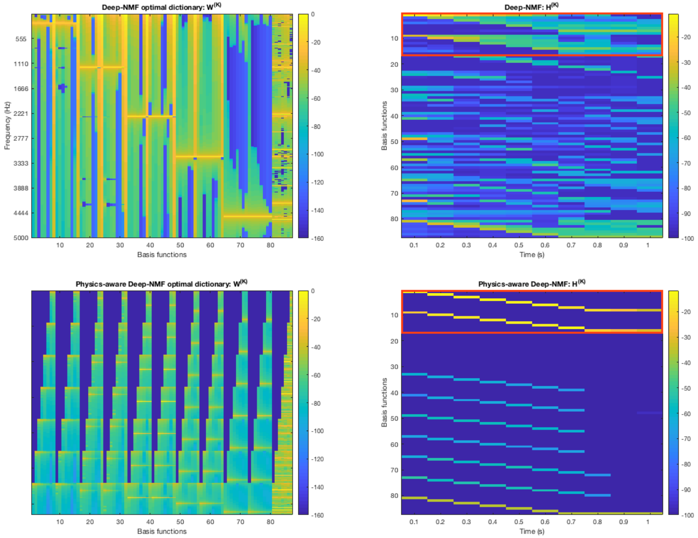
Penalty term

$$\mathcal{E} = \|S - F \circ X\|_2^2 + \gamma \|S - W_S^{(K)} H_S^{(K)}\|_2^2$$

Wiener filter term

[2] Erik Chinellato and Fabio Marcuzzi. Hits detection in audio mixtures by means of a physics-aware Deep-NMF algorithm, in review, 2024.

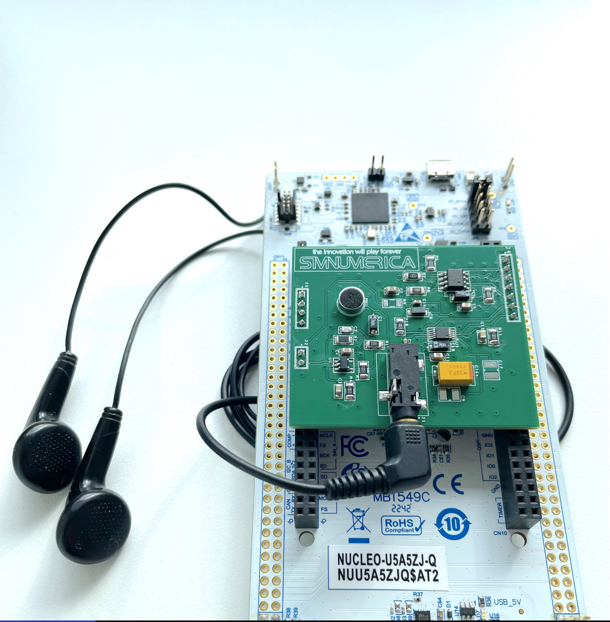
Physics-aware soft-sensor III: hits detection from audio source-separation



Physics-aware soft-sensor III: hits detection from audio source-separation

In order to process 1 second of mixture signal and produce the Physics-aware Deep-NMF output matrix $H^{(K)}$, for the above example around 5.5 million floating point operations are needed. As a consequence, an **ARM4 microcontroller** (144 MHz, with a HW single-precision floating-point unit) with its $\sim$ 8MFLOPS, can **run the complete algorithm** in less than 0.7 **seconds**.

The IOT system



Computing resources vs computational cost

Denoting:

- M: total number of frequency buckets
- N: time instants resulting from applying the STFT to 1 sec long signal mixtures,
- K: the number of layers in the network,
- R: the number of columns in each PAD-NMF dictionary $W^{(k)}$,
- T: the number of Hankel concatenations,

the computational cost of a forward run of the PAD-NMF algorithm is:

$$C_{\text{PAD}-\text{NMF}} = O\left(MTR + (K-1)\left(MTRN + MTN + 2RN + 1\right)\right)$$

Note that $(M, N) = \text{size}(X)$.

Computing resources vs computational cost

Furthermore, letting N_S be the number of sources we are trying to detect and reconstruct, $R_n\ \forall n = 1, \ldots, N_S$ the number of columns allocated within the dictionaries to describe the corresponding source, and M_{IT} the number of optimization iterations employed to solve problem (**??**), the computational cost of the hit detection procedure is:

$$C_{\mathrm{HitDet}} = O\left(N_S(6N_S+7)M_{\mathrm{IT}} + N_S + \sum_{n=1}^{N_S}(R_n+N-1)(R_n+1)\right)$$

Lastly, denoting n_X the number of samples of the discrete-time mixture signals (for 1s mixtures $n_X = f_s$ where f_s is the sampling frequency) and W the window length used in the STFT expressed in number of samples, the clean audio signal generation has a computational cost given by:

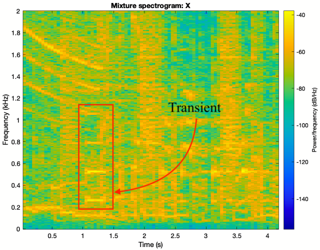
$$C_{\mathrm{SigGen}} = O\left(n_X + 2N(W+1)\sum_{n=1}^{n_S} R_n\right)$$

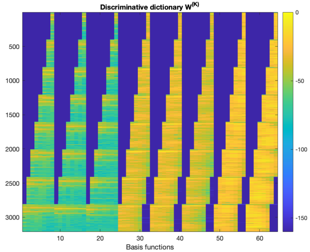
To be run online, execution time has to remain below 1s. Assuming $N_S = 3$ and $n_X = 44100$:

An ARM4 micro controller, given its $\sim$ 8MFLOPS, allows us to let $M = 143$, $N = T = 5$, $K = 4$, $R = 55$, $R_n \equiv 5$, $M_{\mathrm{IT}} = 5000$ and $W = 22050$ for a total execution time of 0.8641s.

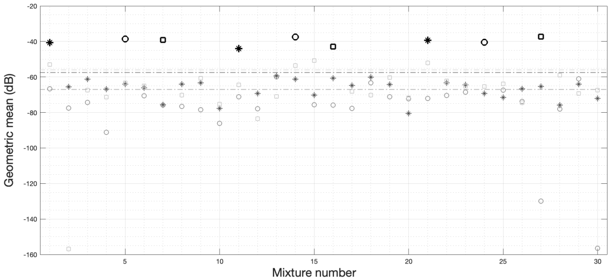
An ARM7 micro controller, given its $\sim$ 20MFLOPS, allows us to let $M = 143$, $N = T = 8$, $K = 4$, $R = 88$, $R_n \equiv 8$, $M_{\mathrm{IT}} = 5000$ and $W = 17640$ for a total execution time of 0.8495s.

Experiment: piano notes and a loud music



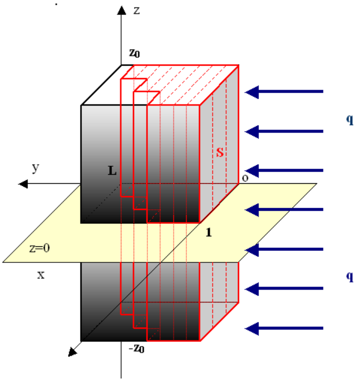


Experiment: piano notes and a loud music



[2] E. Chinellato, F. Marcuzzi, P. Martin. Real-Time Generation of a Targeted Clean Audio Sequence from Source Separation of Noisy Environmental Mixtures Using a Deep Nonnegative Matrix Factorization on IOT Devices. ICT for Intelligent Systems. ICTIS 2024. Smart Innovation, Systems and Technologies, vol 404. Springer, Singapore. https://doi.org/10.1007/978-981-97-5810-4_3.

Physics-aware soft-sensor IV: inner voids/material changes estimation from thermographic inspection



Aim: to estimate the inner material change without solving a nonlinear shape estimation problem.

We demonstrated[1] that there exists a fictitious heat source $f_\vartheta(t,x,y)$

such that for the solution $T_{(f_\vartheta)}$ of

$$\begin{cases} \rho C\ _t T_{(f_\vartheta)} = \kappa\,\Delta T_{(f_\vartheta)} + f_\vartheta, & in\ D_c^{(0)} \times [0,t_f] \\ \kappa\,\nabla T_{(f_\vartheta)} \cdot \mathbf{n}_S = q(t), & on\ S \times [0,t_f] \\ \kappa\,\nabla T_{(f_\vartheta)} \cdot \mathbf{n} = 0, & on\ \partial D_c^{(0)}/S \times [0,t_f] \\ T_{(f_\vartheta)}(0,\cdot) = T_0(\cdot), & in\ D_c^{(0)}. \end{cases} \tag{0.1}$$

it holds that $T_{(f_\vartheta)}(t,x,y) = T^{(\vartheta)}(t,x,y)$ for all $(x,y) \in S$ and for all $t \in [0,t_f]$ and $\mathbf{f}_\theta$ **has compact support within the region corresponding to the void**.

Then, our inverse problem becomes to estimate the fictitious heat source $f_\vartheta(t,x,y)$ in (0.1).

[1] Giusteri GG, Marcuzzi F, Rinaldi L. Replacing voids and localized parameter changes with fictitious forcing terms in boundary-value problems, Results in Applied Mathematics, **20**, 2023, 10.1016/j.rinam.2023.100402

Physics-aware soft-sensor IV: inner voids/material changes estimation from thermographic inspection

In [1] the source term estimator f^k_{mean} be derived from the PDE equation as

$$f^k_{mean} = \frac{1}{N_t} \sum_t \left(\rho C_t \mathbf{e}_{\theta k} + \kappa_x^2 \mathbf{e}_{\theta k} + \kappa_y^2 \mathbf{e}_{\theta k} \right), \tag{0.2}$$

where derivatives are approximated with a suitable finite-difference scheme using the measured values and we assume to accept the approximation ${}^{2}_{y}\mathbf{e}_{\theta k} \approx 0$ because it cannot be computed directly.

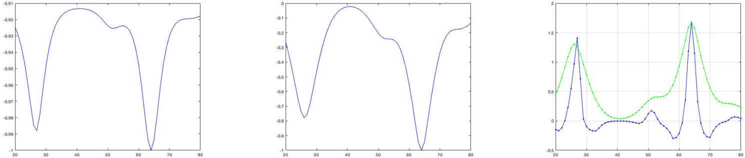


Figure: On the left and on the center, respectively, the prediction error with opposite sign $-\mathbf{e}_{\theta k}$ at the first iteration of the sparse test problem, whose pattern is shown on the left, with a lower and a bigger value of the conductivity coefficient κ. On the right the same prediction error $\mathbf{e}_{\theta k}$ with the higher value of κ (light line) together with the external source estimate obtained with (0.2) (dark line).

[1] Dessole M., Marcuzzi F., Accurate detection of hidden material changes as fictitious heat sources, Numerical Heat Transfer, Part B: Fundamentals, 2023, doi 10.1080/10407790.2023.2220905

Physics-aware soft-sensor IV: inner voids/material changes estimation from thermographic inspection

The fictitious heat source can be estimated from a dynamical system in the so-called *state-space* form:

$$x(k+1) = A \cdot x(k) + B \cdot u(k) + v(k) \quad (0.3)$$

$$y(k) = C \cdot x(k) + D \cdot u(k) + w(k) \quad (0.4)$$

where $\{x(k)\}$ is the state vector, $\{y(k)\}$ is the vector of measured outputs, $\{u(k)\}$ is the vector of inputs (which are supposed known), $\{v(k)\}$ and $\{w(k)\}$ are *model noise* and *measurement noise*, supposed gaussian, with zero mean and covariance matrices Q and R.
Then, under certain assumptions, it is possible to apply the Kalman Filter to estimate the state trajectory of the real system. Let us recall here the KF in its one-step version:

$$P(k) = \left[\left(Q(k-1) + A'(k-1)P(k-1)A'(k-1)^T \right)^{-1} + C^T R^{-1} C \right]^{-1} \quad (0.5)$$

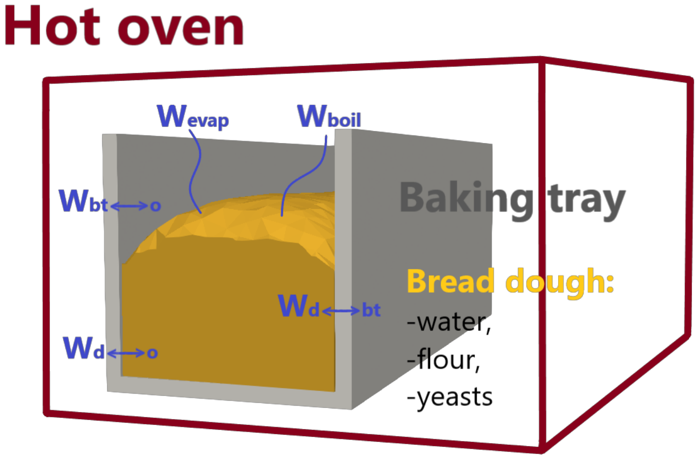
$$\delta\hat{x}(k) = -P(k)\, C^T R^{-1} \left[C \left(A'(k-1)\, \hat{x}(k-1) + B\, u(k-1) \right) - \bar{y}(k) \right] \quad (0.6)$$

$$\hat{x}(k) = A'(k-1)\, \hat{x}(k-1) + B\, u(k-1) + \delta\hat{x}(k) \quad (0.7)$$

Here the scientific machine learning can learn the covariance matrices and avoid to compute the Kalman gain[1].

[1] Guy Revach, Nir Shlezinger, Xiaoyong Ni, A.L. Escoriza, Ruud J. G. Van Sloun and Yonina C. Eldar. KalmanNet: Neural Network Aided Kalman Filtering for Partially Known Dynamics, IEEE Transactions on Signal Processing, 70, pp.1532-1547, 2022

A finite-element computational model for bread leavening and cooking will be presented. The partial differential equations that describe heat exchange and the presence of moisture, yeast and carbon dioxide are coupled with the quasi-static evolution of the growing elastic dough.



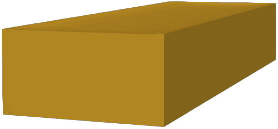
Elasticity with volumetric growth

The dough is treated as a hyperelastic material so, under the action of a body force B, it has an elastic behavior with energy:

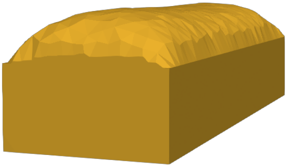
$$E(u) = \int_\Omega \frac{\square}{2}(\operatorname{tr}\mathsf{C} - 3\sqrt[3]{J_{ref}^2}) + \frac{\lambda}{2}(\log(J) - \log(J_{ref}))^2 dX - \int_\Omega B \cdot u \, dX \qquad (0.8)$$

where $\mathsf{F} = \nabla_X u$, $\mathsf{C} = \mathsf{F}^t\mathsf{F}$ and $J = \det \mathsf{F}$.

$$\text{Elasticity equation:} \qquad \delta E(u) = 0 \qquad (0.9)$$



$J_{ref} = 1$



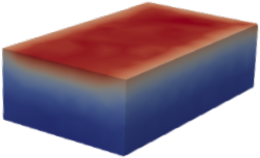
$J_{ref} > 1$

Heat equation in the bread

The temperature of the dough θ evolves according to the heat equation written in material coordinates:

$$\int_{\Omega} \rho\, c_p J \frac{\theta}{t} \tilde{\theta} dX + \int_{\Omega} (k J\, \mathsf{C}^{-1} \nabla_X \theta) \cdot (\nabla_X \tilde{\theta}) dX = \\ = W_{o \longrightarrow d} + W_{bt \longrightarrow d} + W_{loss} \tag{0.10}$$

where $\tilde{\theta}$ is the test function and:



$$W_{o \longrightarrow d} = \int_{\Gamma_1} h_{od} (\theta_{out} - \theta) \tilde{\theta} ds$$

$$W_{bt \longrightarrow d} = \int_{\Gamma_2} h_{bd} (\theta_{bt} - \theta) \tilde{\theta} ds$$

$$W_{loss} = \begin{cases} W_{evap} = - \int_{\Gamma_1} c_{cla} k_2 (\rho_m - \rho_{out}) \tilde{\theta} ds & \text{if } \theta < 100\,^\circ C \\ W_{boil} = - \int_{\Gamma_1} c_p k_3 (\theta_{out} - 100) \rho_m \tilde{\theta} ds & \text{if } \theta \geq 100\,^\circ C \end{cases}$$

Heat equation in the baking tray

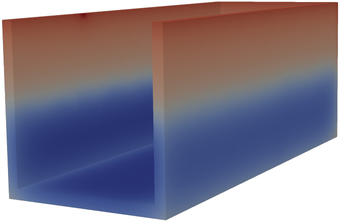
The temperature of the baking tray θ_{bt} changes according to the following heat equation:

$$\int_{\Omega_{bt}} \rho_{bt}\, c_{bt} \frac{\theta_{bt}}{t} \tilde{\theta}_{bt} dX + \int_{\Omega_{bt}} (k_{bt} \nabla_X \theta_{bt}) \cdot (\nabla_X \tilde{\theta}_{bt}) dX = \\ = W_{o \longrightarrow bt} + W_{d \longrightarrow bt} \tag{0.11}$$

where $\tilde{\theta}_{bt}$ is the test function and:

$$W_{o \longrightarrow bt} = \int_{\Gamma_{bt_1}} h_{ob} (\theta_{out} - \theta_{bt}) \tilde{\theta}_{bt} ds$$

$$W_{d \longrightarrow bt} = \int_{\Gamma_{bt_2}} h_{db} (\theta - \theta_{bt}) \tilde{\theta}_{bt} ds$$



Moisture diffusion equation

The water density with respect to the volume of the dough ρ_m, has a diffusive evolution as follows:

$$\int_\Omega J \frac{\rho_m}{t} \tilde{\rho}_m dX + \int_\Omega (\beta J \, \mathsf{C}^{-1} \nabla_X \rho_m) \cdot (\nabla_X \tilde{\rho}_m) dX = M_{loss} \tag{0.12}$$

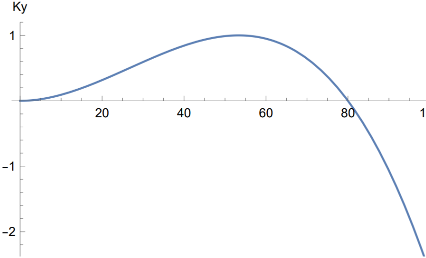
where $\tilde{\rho}_m$ is the test function and:

$$M_{loss} = \begin{cases} M_{evap} = -\int_{\Gamma_1} k_2 (\rho_m - \rho_{out}) \tilde{\rho}_m ds & \text{if } \theta < 100\,^\circ C \\ M_{boil} = -\int_{\Gamma_1} k_3 (\theta_{out} - 100) \rho_m \tilde{\rho}_m ds & \text{if } \theta \geq 100\,^\circ C \end{cases}$$

Rate equation for the yeast evolution

It has been supposed that the yeasts are uniformly distributed in the dough.

Their concentration with respect to the flour quantity Y evolves in time, according to the temperature θ:



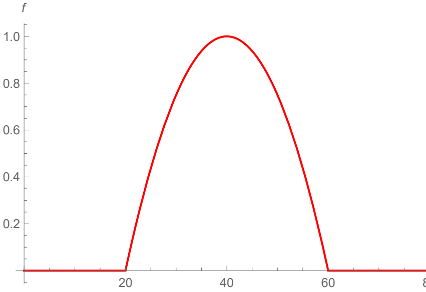
$$\begin{cases} \frac{d}{dt} Y = K_y(\theta) Y & \Longrightarrow \quad Y(t) = \exp(\, K_y(\theta)\, t) Y_0 \\ Y(0) = Y_0 \end{cases} \tag{0.13}$$

Y_0 is the initial concentration and the growth rate is:

$$K_y(\theta) = y_{cost} \frac{(80 - \theta) 27}{80^3} \frac{\theta^2}{4}.$$

CO_2 production and diffusion

The metabolism of the yeasts implies a CO_2 production, so there is an evolution of its concentration D:



$$\int_\Omega J\,\frac{D}{t}\tilde{D}dX + \int_\Omega (\beta_{co2} J\,\mathsf{C}^{-1}\nabla_X D)\cdot(\nabla_X \tilde{D})dX = \int_\Omega f(\theta)\;Y\;\tilde{D}dX \qquad (0.14)$$

where $\tilde{D}$ is the test function and the CO_2 production rate is:

$$f(\theta) = \max\left(-D_{co2}\frac{(\theta-20)(\theta-60)}{400}, 0\right).$$

Volume expansion

The rate of CO_2 involves a volume expansion because of the leavening. From the ideal gas law:

$$Vol_D = \frac{\theta\, n_{mol}\, R}{P_a} \tag{0.15}$$

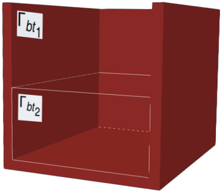
$$J_{ref} = 1 + \frac{Vol_D\, D}{Q} \tag{0.16}$$

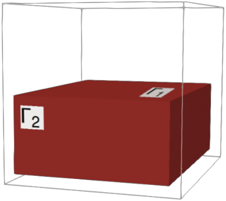
where Vol_D and Q are the volume of one gram of CO_2 and of one gram of dough respectively.

Also the other elastic stiffness change according to the CO_2 quantity, so (at every 10 steps) the elasticity equation is rerun with these updated parameters in order to obtain the volume expansion due to the current CO_2 value.

The meshes

In order to approach the numerical model we construct the geometries and define the meshes on them by choosing the resolution. We divided the boundary of the baking tray and of the dough to manage the heat flux between the two.





Γ_2 and Γ_{bt_2} are the parts of boundary of bread in contact with the baking tray. Γ_1 and Γ_{bt_1} are in contact with the air in the oven.

Since we use material coordinates the domains are fixed, independently of the deformations.

The spaces of finite elements

The spaces of finite elements are defined to the end of approximating functions with polynomials:

Space of functions	**Functions approximated**
V_h is a Vector Function Space defined on meshbr of degree-1 polynomials.	Placement u.
W_h is a Function Space defined on meshbr of degree-2 polynomials.	Temperature of the dough θ, density ρ, moisture yeasts rate Y and CO_2 concentration D.
U_h is a Function Space defined on meshbt of degree-1 polynomials.	Temperature of the baking tray θ_{bt}.

Time evolution

A semi-implicit Euler method is used to deal with the coupling among the different equations

for n in range(n_{steps}):

$\longrightarrow$ solve the heat eq. in the **bread dough**

$\longrightarrow$ solve the diffusion eq. for the **moisture**

$\longrightarrow$ solve the heat eq. in the **baking tray**

$\longrightarrow$ compute the **yeasts** growth rate

$\longrightarrow$ solve the diffusion eq. for the CO_2 **concentration**

$\longrightarrow$ compute the volume occupied by CO_2

if $remainder(n+1, 10) = 0$:

$\longrightarrow$ update the value of J_{ref} and the elastic stiffness

$\longrightarrow$ solve the **elasticity** eq. (iterative nonlinear solver).

end if

end for

The simulations

The implementations using the Python library FEniCS give us the following simulations:

The volume expansion

The partial differential equations that describe heat exchange are coupled with the quasi-static evolution of the growing elastic dough.

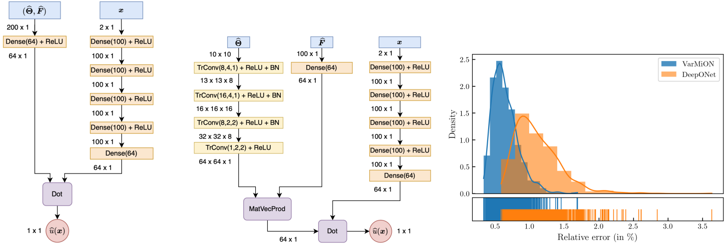
The baking tray temperature evolution

The bread temperature evolution

The baking process

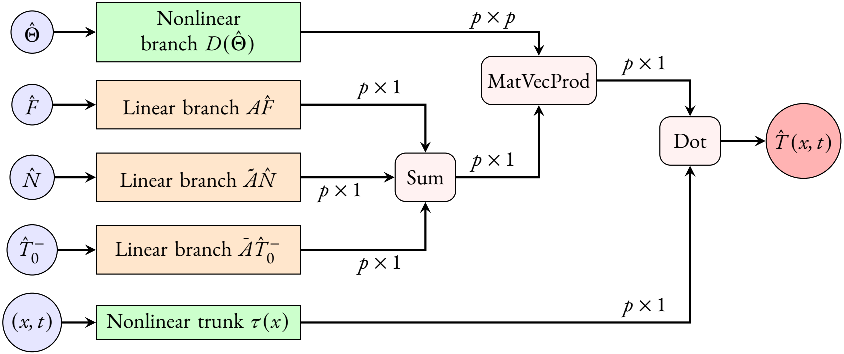
Physics-aware soft-sensor IV: inner voids/material changes estimation from thermographic inspection

Surrogate modelling by scientific machine learning: the most interpretable methodgets best results in force of an architecture that is more tight to the weak form used in Finite Element models.



An ARM4 microcontroller (144 MHz, with a HW single-precision floating-point unit) with its ~ 8MFLOPS, can compute the predicted temperature in a single-point in space in 0.09 seconds.

Physics-aware soft-sensor IV: inner voids/material changes estimation from thermographic inspection



Deep Kalman Filter

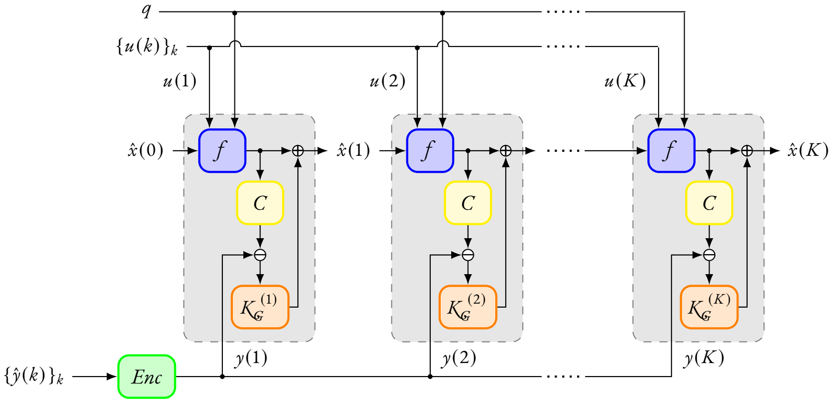


Figure: Structure of the Deep Kalman Filter network.

[2] E. Chinellato, F. Marcuzzi, The Deep Kalman Filter, Springer LNCS 14836, Computational Science – ICCS 2024, Part V, Chapter 22 (2024) 307–321doi:https://doi.org/10.1007/978-3-031-63775-9.

[3] E. Chinellato, Fabio Marcuzzi. State, parameters and hidden dynamics estimation with the Deep Kalman Filter: regularization strategies, submitted, 2024.

Physics-aware soft-sensor IV: inner voids/material changes estimation from thermographic inspection

Thank You for your attention!

questions? → `marcuzzi@math.unipd.it`

Source code will be available at the NLALDlab repository: `https://github.com/NLALDlab.`