Bayesian data-driven model discovery under uncertainty

M. Ceoloni, F. Fatone, F. Fedeli

Supervised by:
Prof. A. Guglielmi, Prof. A. Manzoni

January 8, 2021

A brief recap...

- **Model identification** and **parameters estimation** are of extreme importance in multiple fields of application.
- We started approaching these two problems relying on the approach presented in Yang, Bhouri, Perdikaris (2020), introducing a **Bayesian framework** for uncertainty quantification and taking advantage of **Hamiltonian Monte Carlo (HMC)** sampling.

IL FAMOSO RO

Coronavirus, ecco regione per regione l'indice di trasmissibilità della malattia

L' Ro indica il numero di infezioni prodotte da una persona nell'arco del suo periodo infettivo. Un dato associato al tempo che intercorre nel passaggio della malattia fra un infetto primario e quelli secondari

The original framework [Yang, Bhouri, Perdikaris (2020)]

Dynamical system $\dot{\boldsymbol{x}} = f(\mathbf{x}, t; \boldsymbol{\theta})$ and Bayesian (Lasso-like) model

$$p(\boldsymbol{x}(t+\Delta t)|\boldsymbol{x}(t), \boldsymbol{\theta}, \gamma) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}(t_i + \Delta t_i); h_{\boldsymbol{\theta}}, \gamma^{-1})$$

$$p(\boldsymbol{\theta}|\lambda) = Laplace(\boldsymbol{\theta}; \lambda^{-1})$$

$$p(log\lambda) = Gamma(log\lambda; n_{par} + 1, 1)$$

$$p(log\gamma) = Gamma(log\gamma; 1, 1)$$

$$\Longrightarrow \mathrm{p}(\gamma, \lambda, \boldsymbol{\theta}|\mathbf{x}(t+\Delta t), \mathbf{X}(t)) \propto p(\mathbf{x}(t+\Delta t)|\mathbf{X}(t), \boldsymbol{\theta}, \gamma) p(\boldsymbol{\theta}|\lambda) p(\gamma) p(\lambda),$$

$\boldsymbol{\theta}$ parameters, γ precision parameter, λ shrinkage parameter, $h_{\boldsymbol{\theta}}$ numerical solver output (RK4)

The original framework [Yang, Bhouri, Perdikaris (2020)]

Dynamical system $\dot{\boldsymbol{x}} = f(\mathbf{x}, t; \boldsymbol{\theta})$ and Bayesian (Lasso-like) model

$$p(\boldsymbol{x}(t+\Delta t)|\boldsymbol{x}(t), \boldsymbol{\theta}, \gamma) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}(t_i+\Delta t_i); h_{\boldsymbol{\theta}}, \gamma^{-1})$$

$$p(\boldsymbol{\theta}|\lambda) = Laplace(\boldsymbol{\theta}; \lambda^{-1})$$

$$p(log\lambda) = Gamma(log\lambda; n_{par}+1, 1)$$

$$p(log\gamma) = Gamma(log\gamma; 1, 1)$$

$$\Longrightarrow \mathrm{p}(\gamma, \lambda, \boldsymbol{\theta}|\mathbf{x}(t+\Delta t), \mathbf{X}(t)) \propto p(\mathbf{x}(t+\Delta t)|\mathbf{X}(t), \boldsymbol{\theta}, \gamma)p(\boldsymbol{\theta}|\lambda)p(\gamma)p(\lambda),$$

$\boldsymbol{\theta}$ parameters, γ precision parameter, λ shrinkage parameter, $h_{\boldsymbol{\theta}}$ numerical solver output (RK4)

Parameter estimation can be summarized in two steps:

The original framework [Yang, Bhouri, Perdikaris (2020)]

Dynamical system $\dot{\boldsymbol{x}} = f(\mathbf{x}, t; \boldsymbol{\theta})$ and Bayesian (Lasso-like) model

$$p(\boldsymbol{x}(t+\Delta t)|\boldsymbol{x}(t), \boldsymbol{\theta}, \gamma) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}(t_i + \Delta t_i); h_{\boldsymbol{\theta}}, \gamma^{-1})$$

$$p(\boldsymbol{\theta}|\lambda) = Laplace(\boldsymbol{\theta}; \lambda^{-1})$$

$$p(log\lambda) = Gamma(log\lambda; n_{par} + 1, 1)$$

$$p(log\gamma) = Gamma(log\gamma; 1, 1)$$

$$\Longrightarrow \mathrm{p}(\gamma, \lambda, \boldsymbol{\theta}|\mathbf{x}(t+\Delta t), \mathbf{X}(t)) \propto p(\mathbf{x}(t+\Delta t)|\mathbf{X}(t), \boldsymbol{\theta}, \gamma) p(\boldsymbol{\theta}|\lambda) p(\gamma) p(\lambda),$$

$\boldsymbol{\theta}$ parameters, γ precision parameter, λ shrinkage parameter, $h_{\boldsymbol{\theta}}$ numerical solver output (RK4)

Parameter estimation can be summarized in two steps:

1. Numerical (deterministic) estimation of the parameters' starting point for the HMC algorithm and the required gradients

The original framework [Yang, Bhouri, Perdikaris (2020)]

Dynamical system $\dot{\boldsymbol{x}} = f(\mathbf{x}, t; \boldsymbol{\theta})$ and Bayesian (Lasso-like) model

$$p(\boldsymbol{x}(t+\Delta t)|\boldsymbol{x}(t), \boldsymbol{\theta}, \gamma) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}(t_i + \Delta t_i); h_{\boldsymbol{\theta}}, \gamma^{-1})$$

$$p(\boldsymbol{\theta}|\lambda) = Laplace(\boldsymbol{\theta}; \lambda^{-1})$$

$$p(log\lambda) = Gamma(log\lambda; n_{par} + 1, 1)$$

$$p(log\gamma) = Gamma(log\gamma; 1, 1)$$

$$\Longrightarrow \mathrm{p}(\gamma, \lambda, \boldsymbol{\theta}|\mathbf{x}(t+\Delta t), \mathbf{X}(t)) \propto p(\mathbf{x}(t+\Delta t)|\mathbf{X}(t), \boldsymbol{\theta}, \gamma)p(\boldsymbol{\theta}|\lambda)p(\gamma)p(\lambda),$$

$\boldsymbol{\theta}$ parameters, γ precision parameter, λ shrinkage parameter, $h_{\boldsymbol{\theta}}$ numerical solver output (RK4)

Parameter estimation can be summarized in two steps:

1. Numerical (deterministic) estimation of the parameters' starting point for the HMC algorithm and the required gradients
2. Sampling from the posterior through HMC

Original framework - Drawbacks

- **Issues** when working with **real-world data** due to 'batch-feeding' mechanism of the original framework.
- **Differentiation** w.r.t. parameters **is difficult** abandoning the original (TensorFlow) framework.
- **HMC** seemed **oversized** for the dimensions at stake.
- There's actually **no evidence** to take a **Gaussian error** on fitted data w.r.t. reality.

Original framework - Drawbacks

- **Issues** when working with **real-world data** due to 'batch-feeding' mechanism of the original framework.
- **Differentiation** w.r.t. parameters **is difficult** abandoning the original (TensorFlow) framework.
- **HMC** seemed **oversized** for the dimensions at stake.
- There's actually **no evidence** to take a **Gaussian error** on fitted data w.r.t. reality.

We urged to search for a different approach...

We chose the **ABC-SMC scheme** described by Toni et al. (2009):

Algorithm 1: ABC - Sequential Monte Carlo

Result: A sample from $p_\epsilon(\boldsymbol{\theta}|x)$

Initialization: A precision schedule $\{\epsilon_t\}_{t\in 1:T}$;

while $t \leq T$ **do**

  while $n \leq N$ **do**

    if $t = 1$ **then**

      sample $\tilde{\theta}$ from $\pi(\theta)$;

    else

      sample θ from the previous population $\{\theta^{(i,t-1)}\}_i$ with weights $\{\omega^{(i,t-1)}\}_i$;

      sample $\tilde{\theta}$ from $K_t(\cdot|\theta)$ s.t. $\pi(\theta) > 0$;

    end

    compute $y = f(\cdot|\tilde{\theta})$;

    if $\Delta(y, x) \leq \epsilon_t$ **then**

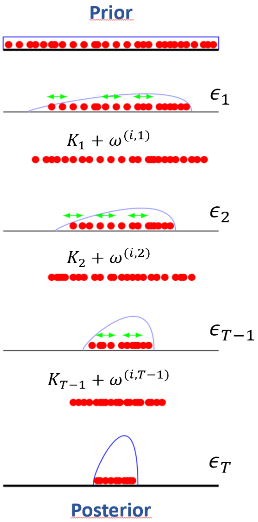
      save $\tilde{\theta}$ and y;

    else

  end

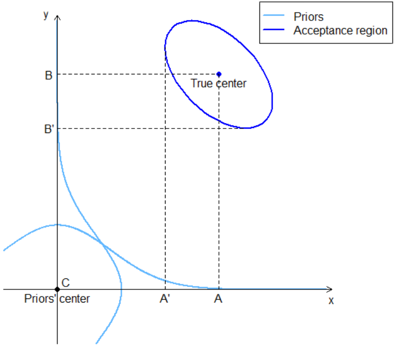
  compute $\{\omega^{(i,t)}\}_i$ and normalize them;

end



ABC-SMC Issues

Standard ABC-SMC algorithm showed **problems** in case of priors being **not diffuse enough**:



Given the prior distribution center C and the 'true' center (A, B), in this case we have that

$$\mathbb{P}(\mathfrak{R}_\epsilon) \leq 3 \cdot 10^{-5}$$

considering $\mathfrak{R}_\epsilon$ the **acceptance region**.

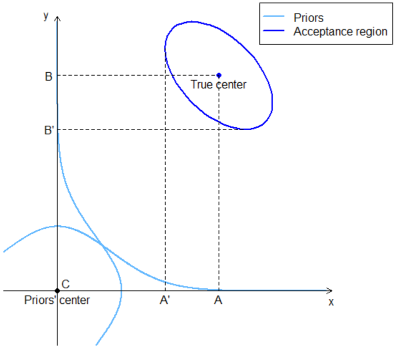
Also, we have that

$$\mathcal{L}(X \in \mathfrak{R}_\epsilon) \propto \mathcal{L}(X \in \mathfrak{R}_\epsilon | X_1 \geq A', X_2 \geq B')$$

thus suggesting a **preconditioning framework**, as we actually know a good estimate of the true center from the numerical estimation phase.

ABC-SMC Preconditioning phase

The preconditioning algorithm we designed is the following one:



Algorithm 2: ABC-SMC 'empirical' preconditioning (1D)

Result: A′: estimate of the border of $\mathfrak{R}_{\epsilon_{start}}$

Initialization: A: estimate of the posterior center, *tol* $\in (0,1)$;

while $n \leq N$ **do**
    if $A > q_\pi(1 - tol)$ *or* $A < q_\pi(tol)$ **then**
        sample P $\sim \mathcal{U}((C, A))$;
        sample $\tilde{\theta}$ from $\pi(\theta|\theta > P)$;
    else
        sample $\tilde{\theta}$ from $\pi(\theta)$;
    end
    compute $y = f(\cdot|\tilde{\theta})$;
    if $\Delta(y, x) \leq \epsilon$ **then**
        save $\tilde{\theta}$;
    else
end

return $min(\tilde{\theta})$

Lotka-Volterra (LV) model

We applied our idea to the LV model:

$$\begin{cases} \dot{x}_1 = \theta_1 x_1 - \theta_2 x_1 x_2 \\ \dot{x}_2 = \theta_3 x_1 x_2 - \theta_4 x_2 \end{cases}$$

Lotka-Volterra (LV) model

We applied our idea to the LV model:

$$\begin{cases} \dot{x}_1 = \theta_1 x_1 - \theta_2 x_1 x_2 \\ \dot{x}_2 = \theta_3 x_1 x_2 - \theta_4 x_2 \end{cases}$$

$$+$$

Ridge-like shrinkage model for the bayesian framework:

$$p(\boldsymbol{\theta}|\lambda) = \mathcal{N}(\boldsymbol{\theta}; 0, \lambda^{-1})$$

$$\mathrm{p}(\log\ \lambda) = Gamma(log\,\lambda; n_{par} + 1, 1)$$

Lotka-Volterra (LV) model

We applied our idea to the LV model:

$$\begin{cases} \dot{x}_1 = \theta_1 x_1 - \theta_2 x_1 x_2 \\ \dot{x}_2 = \theta_3 x_1 x_2 - \theta_4 x_2 \end{cases}$$

$$+$$

Ridge-like shrinkage model for the bayesian framework:

$$p(\boldsymbol{\theta}|\lambda) = \mathcal{N}(\boldsymbol{\theta}; 0, \lambda^{-1})$$

$$\mathrm{p}(\log\,\lambda) = Gamma(log\,\lambda; n_{par} + 1, 1)$$

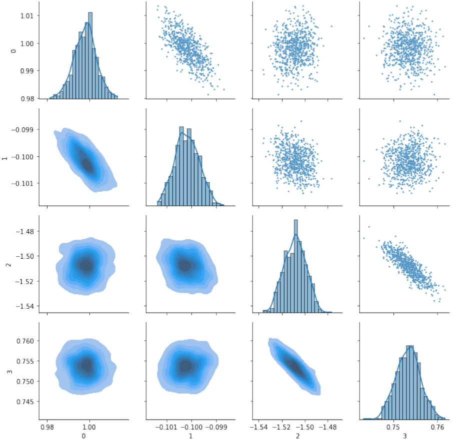
$$+$$

$$\Delta(\boldsymbol{x}, \boldsymbol{y}(\boldsymbol{\theta})) = \textstyle\sum_{i=1}^{n} ||\boldsymbol{x}(t_i) - h_{\boldsymbol{\theta}}(t_i)||^2$$

We considered this problem to compare ABC-SMC vs Perdikaris'performances.

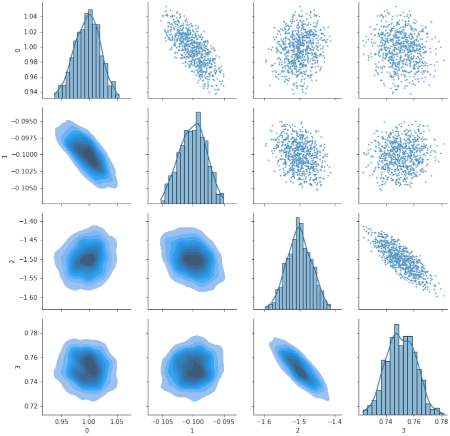
HMC and ABC-SMC comparison

LV using HMC (Perdikaris)



Computational time: 1h 05 min

LV using eABC-SMC

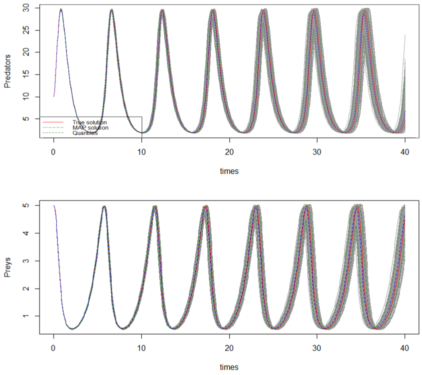


Computational time: 21 min

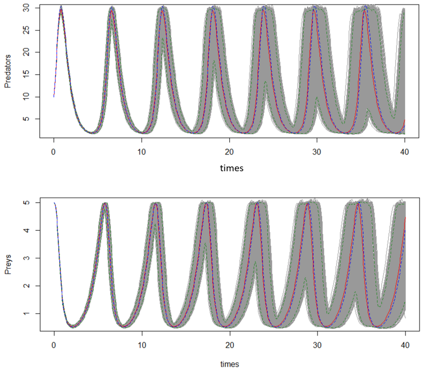
HMC and ABC-SMC comparison

We also considered the confidence bands for the system evolution in time:

LV using HMC (Perdikaris)



LV using eABC-SMC



Observed data ranges from time $t = 0$ to time $t = 20$.

HMC and ABC-SMC comparison

In order to find the point estimates of the parameters we used 4D kernel smoothing to find the Maximum A Posteriori estimate in both HMC and ABC-SMC posterior samples:

$$\boldsymbol{\theta}_{\boldsymbol{MAP}} = arg \max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}, \lambda, \gamma | \mathcal{D})$$

	HMC Estimates				**ABC-SMC Estimates**			
	θ_1	θ_2	θ_3	θ_4	θ_1	θ_2	θ_3	θ_4
True	1	-0.1	-1.5	0.75	1	-0.1	-1.5	0.75
MAP	0.9979039	-0.1003933	-1.506464	0.7529937	0.996897	-0.098455	-1.506832	0.7596772
Rel. error	2.0961e-3	3.933e-3	4.309e-3	3.9916e-3	3.103e-3	1.544e-2	4.55e-3	1.290e-2

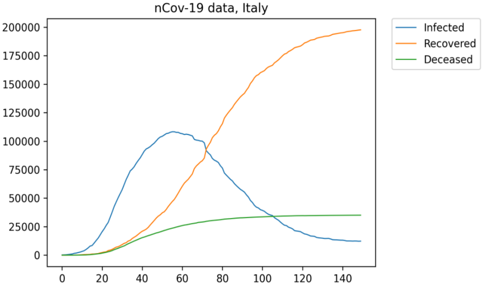
Furthermore, we investigated the collinearity relationship among θ_1 - θ_2 and θ_3 - θ_4, finding the following promising results:

	$\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2$	$\boldsymbol{\theta}_3 - \boldsymbol{\theta}_4$
Slope HMC	-7.831	-2.9021
Slope ABC	-6.842	-2.6087
$\boldsymbol{R^2}$ **HMC**	0.4901	0.7324
$\boldsymbol{R^2}$ **ABC**	0.5078	0.6731

Therefore, besides the qualitative resemblance between the two posterior distributions, we found evidence of a more quantitative similarity.

SIR model

After good results with LV were obtained, we moved on to our initial goal of trying to identify parameters in epidemiological models (SIR, SEIRD etc) from real nCov-19 data.



t	date
0	February 20th
10	March 1st
18	March 9th (Lockdown start)
41	April 1st
71	May 1st
88	May 18th (Lockdown end)

Data: Protezione Civile

This, however, entails some practical problems, mainly arising because of the initial underreporting and the time-varying behavior of the parameters.

Final goals

- Use our **ABC-SMC** framework with **real nCov-19 data**, working with **complex epidemiological models** with non-observable compartments.
- Exploit other countries' data (e.g. China) to build **more informative priors** for the model.
- Model parameters' dynamical behavior with a **time-varying parametric form**.
- **Build an accurate forecasting framework** using different countries' data with **meaningful uncertainty quantification** on the predictions.

Thank You

Y. Yang, M.A. Bhouri, P. Perdikaris (2020). Bayesian differential programming for robust systems identification under uncertainty. ArXiv pre-print, submitted to Proceedings of the Royal Society A.

Toni, T., Welch, D., Strelkowa, N., Ipsen, A., Stumpf, M. P. (2009). Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. Journal of the Royal Society Interface, 6(31), 187-202.

https://github.com/CSSEGISandData/COVID-19˙Unified-Dataset

Appendix 1 - Weights for the ABC-SMC [Toni et al., 2009]

Weights used to implement ABC-SMC algorithm are the following:

For $t \neq 1$

$$\omega^{(i,t)} = \frac{\pi(\theta^{(i,t)})}{\sum_{j=1}^{n} \omega^{(j,t-1)} K_t(\theta^{(i,t)} | \theta^{(j,t-1)})}$$

Furthermore, we used a Gaussian kernel, described as follows:

$$K_t(\theta_k | \theta_k^{(j,t-1)}) = \frac{1}{\sqrt{2\pi\sigma_{(k,t)}^2}} e^{\frac{-(\theta_k - \theta_k^{(j,t-1)})^2}{2\sigma_{(k,t)}^2}}$$

Common choice in literature (Toni et al., 2009, Filippi et al. 2012)

Appendix 2 - Complete eABC-SMC Algorithm

Algorithm 3: eABC-SMC

Result: A sample from $p_\epsilon(\boldsymbol{\theta}|x)$
Initialization: A precision schedule $\{\epsilon_t\}_{t\in 1:T}$, the estimated true center C ;
Preconditioning Estimate the borders through **Algorithm 2**;
while $t \leq T$ **do**
    while $n \leq N$ **do**
        if $t = 1$ **then**
            sample $\tilde{\theta}$ from $\pi(\theta|\mathfrak{R}_{bor})$, with $\mathfrak{R}_{bor}$ being the region delimited by borders;
        else
            sample θ from the previous population $\{\theta^{(i,t-1)}\}_i$ with weights $\{\omega^{(i,t-1)}\}_i$;
            sample $\tilde{\theta}$ from $K_t(\cdot|\theta)$ s.t. $\pi(\theta) > 0$;
        end
        compute $y = f(\cdot|\tilde{\theta})$;
        if $\Delta(y, x) \leq \epsilon_t$ **then**
            save $\tilde{\theta}$ and y;
        else
    end
    compute $\{\omega^{(i,t)}\}_i$ and normalize them;
end