Adventures in Gaussian Processes

AI4Research seminar, Uppsala University

Mattias Villani

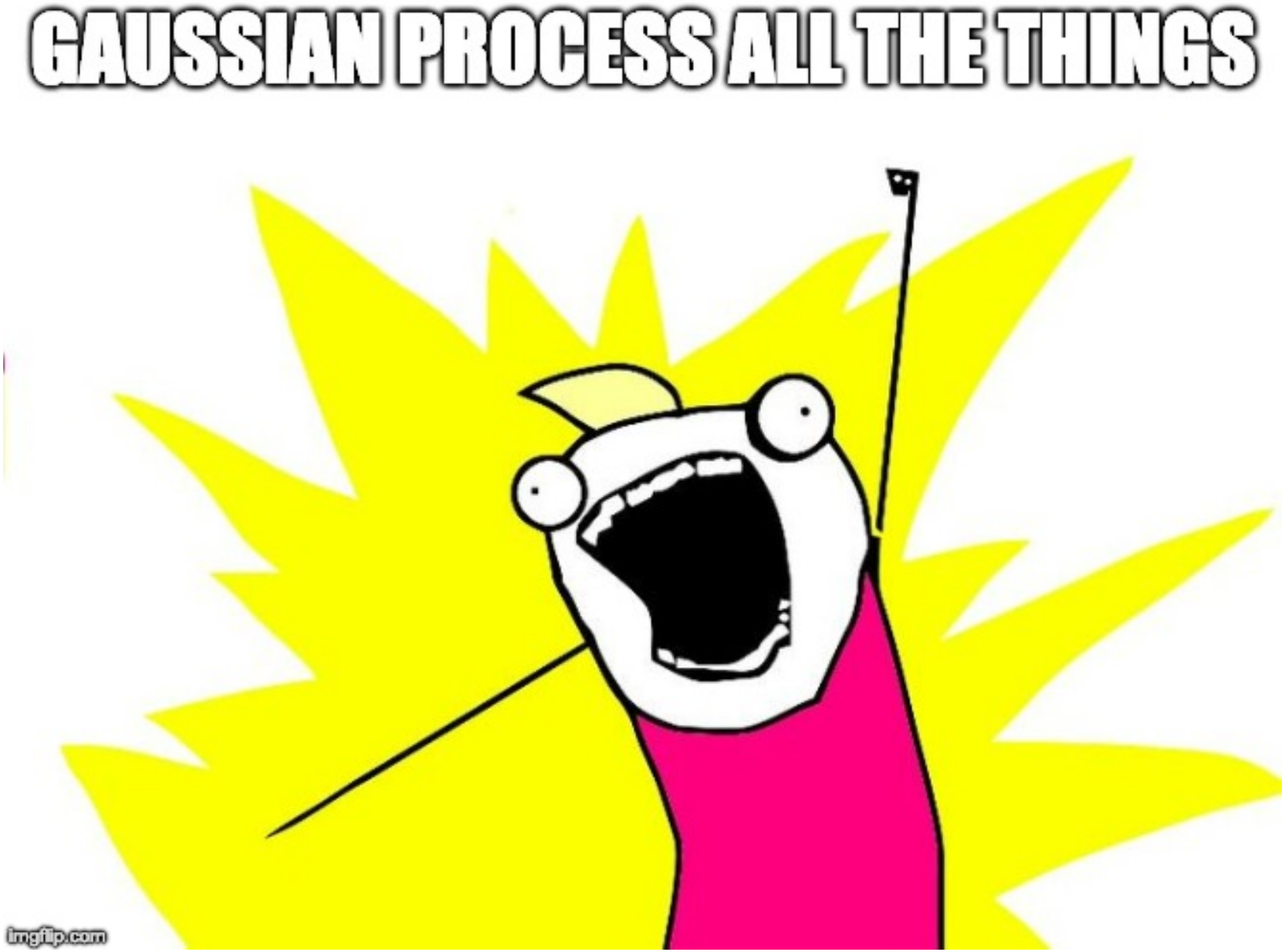
Department of Statistics
Stockholm University

Department of Computer and Information Science
Linköping University

Overview

- **Gaussian processes** for data analysis - brief intro.
- **Search-and-rescue** using hierarchical spatial point processes.
- **Dynamic multi-layer network** modeling and prediction of flight connections.
- **[Bayesian optimization** with user-controlled precision.**]**

Alternative title



Gaussian process regression

Gaussian process regression

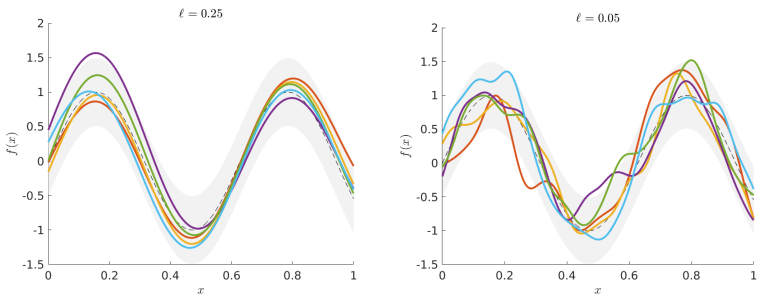
$$y_i = f(x_i) + \varepsilon_i, \qquad \varepsilon_i \overset{\text{iid}}{\sim} N(0, \sigma_n^2)$$

Gaussian process prior over the space of functions

$$f(x) \sim \mathcal{GP}\left(m(x), k(x, x')\right)$$

with **squared exponential** covariance function

$$k(x, x') \equiv \text{Cov}(f(x), f(x')) = \sigma_f^2 \exp\left(-\frac{\|x - x'\|^2}{2\ell^2}\right)$$



Gaussian process regression

- **Model**

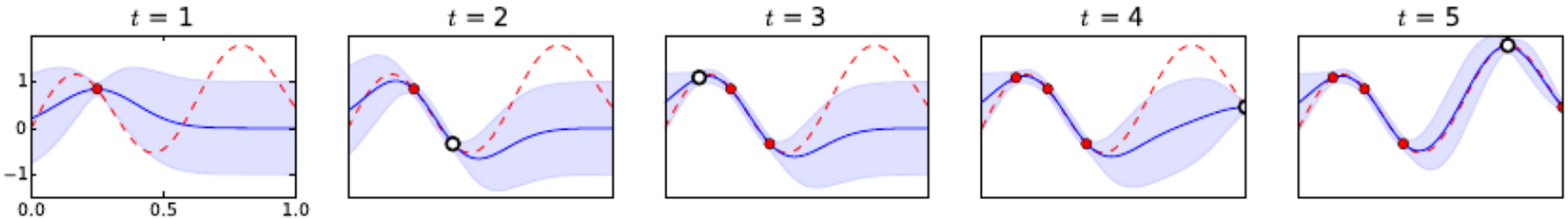
$$y_i = f(\boldsymbol{x}_i) + \varepsilon_i, \quad \varepsilon \overset{iid}{\sim} N(0, \sigma_n^2)$$

- **Prior**

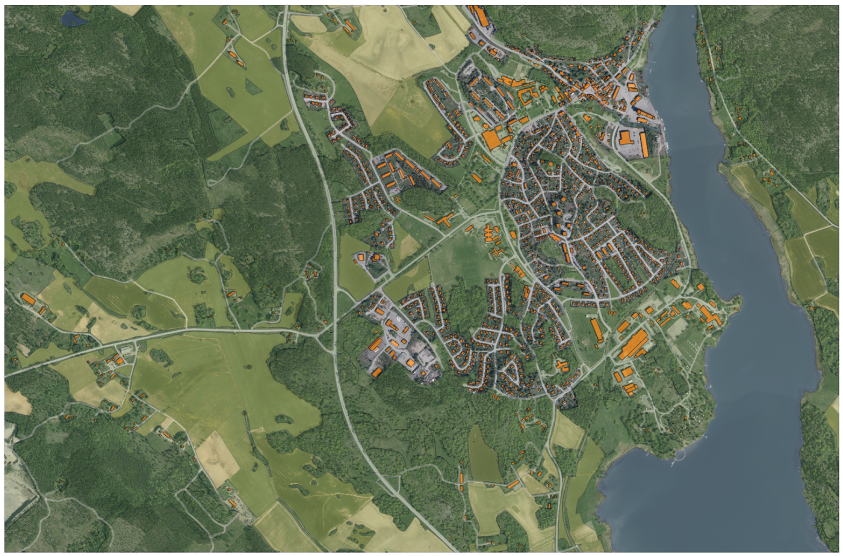
$$f(\boldsymbol{x}) \sim GP\left(0, k(\boldsymbol{x}, \boldsymbol{x}')\right)$$

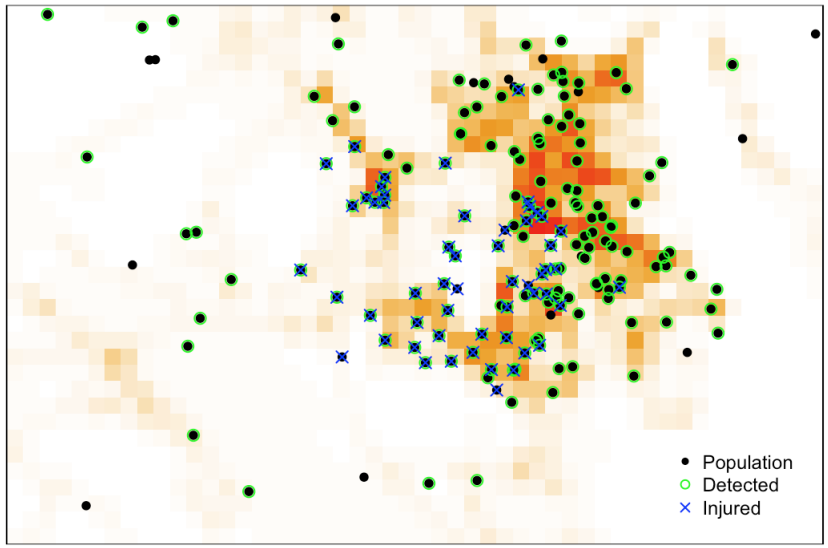
- **Posterior** of $f(\cdot)$ over test data $\boldsymbol{X}_*$:

$$\begin{aligned} \boldsymbol{f}_* | \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_* &\sim N\left(\bar{\boldsymbol{f}}_*, \operatorname{cov}(\bar{\boldsymbol{f}}_*)\right) \\ \bar{\boldsymbol{f}}_* &= K(\boldsymbol{X}_*, \boldsymbol{X})\left(K(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y} \\ \operatorname{cov}(\bar{\boldsymbol{f}}_*) &= K(\boldsymbol{X}_*, \boldsymbol{X}_*) - K(\boldsymbol{X}_*, \boldsymbol{X})\left(K(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}\right)^{-1} K(\boldsymbol{X}, \boldsymbol{X}_*) \end{aligned}$$



Search-and-rescue using spatial point processes [1]





[1] Andersson et al (2019). Real-Time Robotic Search using Structural Spatial Point Processes. *UAI*.

Hierarchical Spatial Point Process

- **Log Gaussian Cox Process** (LGCP) for **number of persons** in the region $\tilde{S} \subset S$.

$$N_{y^\star}(\tilde{S})|\lambda \sim \text{Poisson}\left(\int_{\boldsymbol{s}\in\tilde{S}} \lambda(\boldsymbol{s})d\boldsymbol{s}\right)$$

$$\log\lambda(\boldsymbol{s}) = \alpha_\lambda + \underbrace{\boldsymbol{x}_\lambda^\top(\boldsymbol{s})}_{GIS}\boldsymbol{\beta}_\lambda + \underbrace{\xi_\lambda(\boldsymbol{s})}_{GP \text{ in } 2D}$$

- The **number of detected** persons by a thinned LGCP

$$N_y(\tilde{S})|r,\lambda \sim \text{Poisson}\left(\int_{s\in\tilde{S}} r(\boldsymbol{s})\lambda(\boldsymbol{s})d\boldsymbol{s}\right)$$

$$\log r(\boldsymbol{s}) = \boldsymbol{x}_r^\top(\boldsymbol{s})\boldsymbol{\beta}_r$$

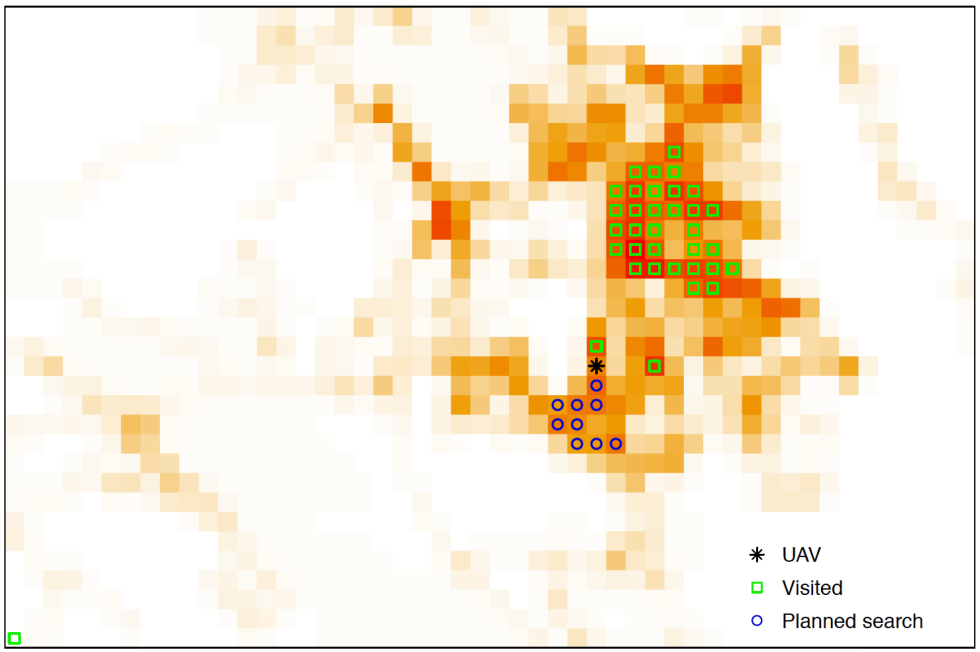
- **Probability of injury** for ith person at location $\boldsymbol{y}_i$

$$w_i|q \sim \text{Bernoulli}\left(q(\boldsymbol{y}_i)\right),$$

$$\log\left(\frac{q(\boldsymbol{s})}{1-q(\boldsymbol{s})}\right) = \alpha_q + \boldsymbol{x}_q^\top(\boldsymbol{s})\boldsymbol{\beta}_q + \xi_q(\boldsymbol{s})$$

Real-time decision making under uncertainty

- Challenges
  1. **missing data** - point pattern is only partially observed
  2. **real-time sequential inference over spatial fields**
  3. **real-time decision making** under uncertainty
- Solutions
  1. Strong priors based on **GIS data**
  2. **Warm-started INLA** for inference
  3. Tailored **Monte Carlo Tree Search** for decision



Discretization of the process

- Partition spatial domain in $a_1 \cdot a_2$ rectangles with area Δ.
- **Number of detected** in region (i, j):

$$n_{ij} \sim \text{Pois}\left(\Delta \exp(z_{\lambda,ij} + z_{r,ij})\right)$$

- Different features in population and detection models.
- **Number of detected injured** in region (i, j):

$$m_{ij}|n_{ij} \sim \text{Binom}\left(n_{ij}, \frac{1}{1 + \exp(-z_{q,ij})}\right)$$

- The latent variables $z_{\lambda,ij}$, $z_{r,ij}$ and $z_{q,ij}$ are representative values of $\log \lambda(\boldsymbol{s})$, $\log r(\boldsymbol{s})$ and $\text{logit}\, q(\boldsymbol{s})$ in region s_{ij}.

Integrated Nested Laplace Approximation (INLA)

- **INLA** targets models of the form

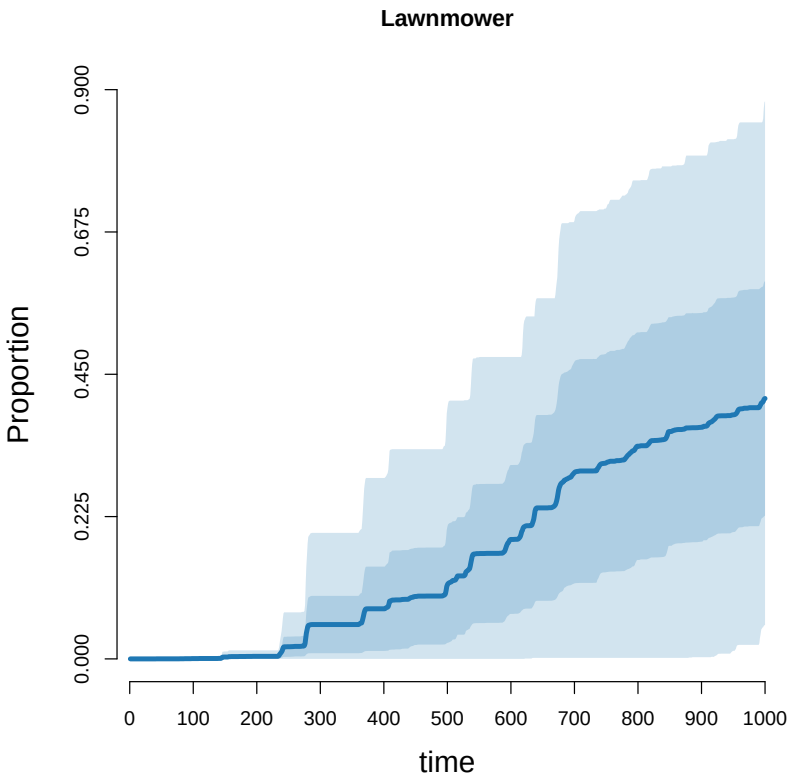
$$\boldsymbol{\theta} \sim \pi(\boldsymbol{\theta})$$
$$\boldsymbol{z}|\boldsymbol{\theta} \sim N\left(0, \boldsymbol{Q}^{-1}(\boldsymbol{\theta})\right), \quad \boldsymbol{Q} \text{ sparse}$$
$$\pi(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{\theta}) = \prod_{i=1}^{n} \pi(y_i|z_i, \boldsymbol{\theta})$$

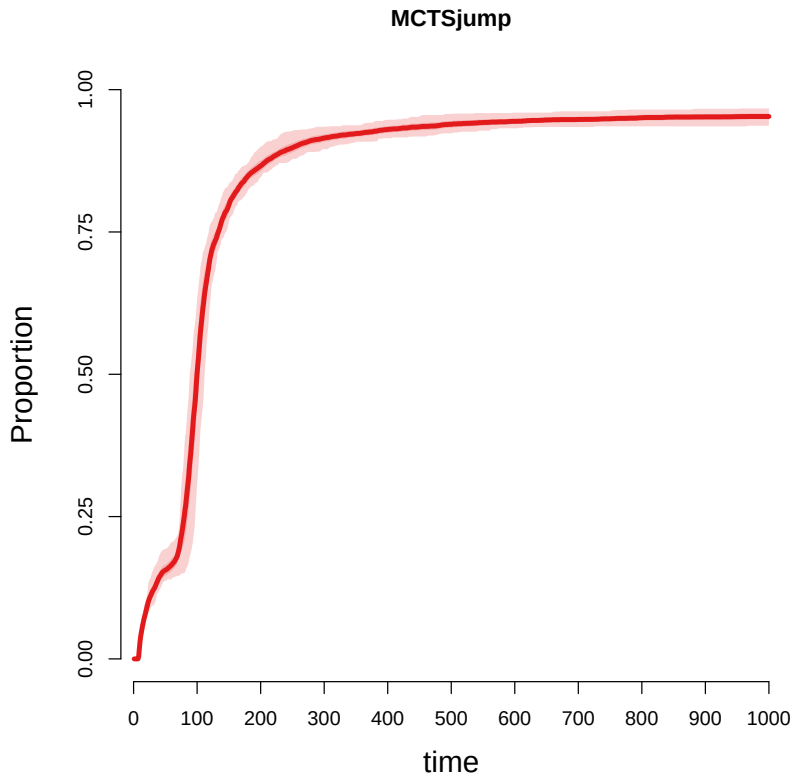
- n is the number of rectangles
- y_i is number of detected or injured in rectangle i
- z_i is $z_{\lambda,ij} + z_{r,ij}$ or $z_{q,ij}$ in rectangle i (+ fixed effects α and β)
- $\boldsymbol{\theta}$ is a low-dim vector with kernel hyperparameters
- **INLA approximates marginal posterior of latents**

$$\pi(z_i|\boldsymbol{y}) = \int \tilde{\pi}(z_i|\boldsymbol{\theta}, \boldsymbol{y})\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{\theta}$$

by Laplace/Gaussian approx ($\boldsymbol{z}$) and numerical integration ($\boldsymbol{\theta}$).

We find injured a lot faster than lawnmower





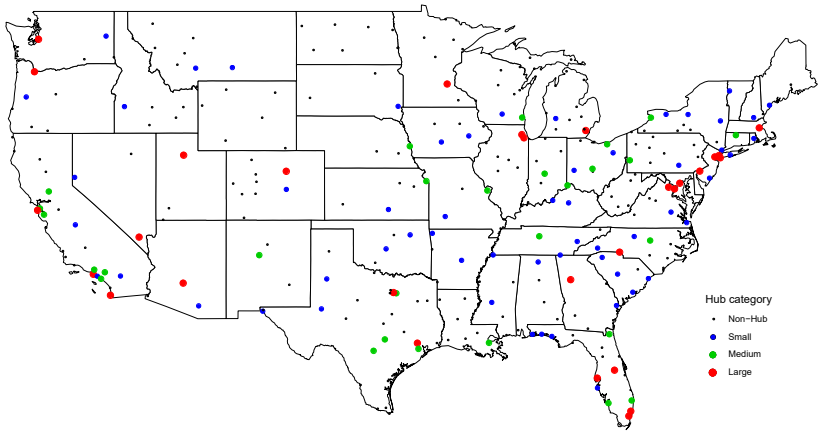
Demo video - finding injured after terrorist attack

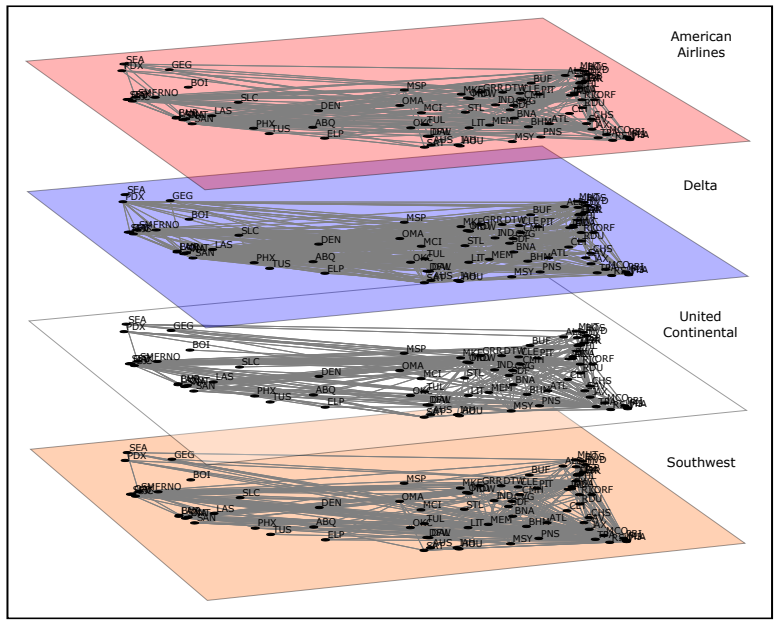
https://www.youtube.com/watch?v=wyD0O5hF5tE&ab_channel=MattiasVillani

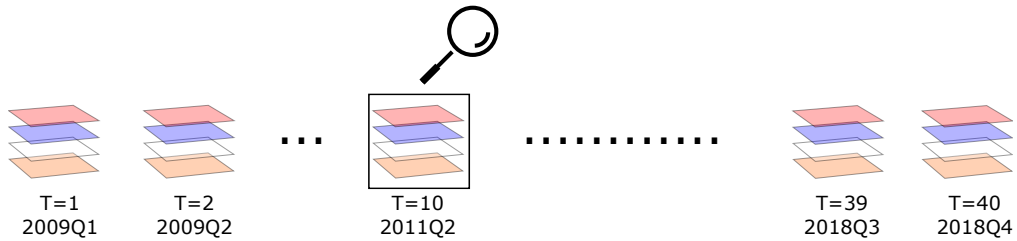
Airline network evolution

- **Aim**: **Predict** the evolution of airline **networks over time**.
- **Data**: Quarterly world-wide networks for american airlines.
- **Model**: **Dynamic multi-layered networks** driven by **GPs**.

American data: 80 airports, 4 airlines, 40 quarters







Dynamic networks driven by latent variables

- **Static Bernoulli model** for adjacency matrix Y

$$Y_{uv}(t) \overset{\text{iid}}{\sim} \text{Bern}(\pi)$$

- **Dynamic Bernoulli** with **global latent GP**

$$Y_{uv}(t)|\pi(t) \overset{\text{iid}}{\sim} \text{Bern}\left(\pi(t)\right)$$
$$\text{Logit}\left[\pi(t)\right] = \log\left(\frac{\pi(t)}{1-\pi(t)}\right) = z(t),$$
$$z(t) \sim \text{GP}(\mu(t), K(t,t'))$$

- **Dynamic Bernoulli** with **latent GPs at nodes**

$$Y_{uv}(t)|(t) \sim \text{Bern}\left[\pi_{uv}(t)\right]$$
$$\text{Logit}\left[\pi_{uv}(t)\right] = z(t) - d\left(x_u(t), x_v(t)\right),$$
$$z(t) \sim \text{GP}(\mu_z(t), K_z(t',t))$$
$$x_u(t) \sim \text{GP}(\mu_u(t), K_u(t',t)),\ u = 1, ..., N.$$

Dynamic multi-layer networks

- **Dynamic multi-layer Bernoulli model**[2] for layer $k = 1, \ldots, K$:

$$Y_{uv}^{(k)}(t) | \pi_{uv}^{(k)}(t) \sim \text{Bern}\left(\pi_{uv}^{(k)}(t)\right)$$

$$\text{Logit}\left[\pi_{uv}^{(k)}(t)\right] = z(t) - d\left(x_u(t), x_v(t)\right) - d\left(x_u^{(k)}(t), x_v^{(k)}(t)\right),$$

  where

  - **Global GP** across airport and airlines: $z(t)$
  - **Airport-specific GPs**, but shared across airlines: $x_u(t)$.
  - **Airport/Airline-specific** GPs: $x_u^{(k)}(t)$.

- Number of GPs: $1 + N + KN$ for N nodes and K layers.
- **Regularization** is important.

[2] Durante et al (2017). Bayesian learning of dynamic multilayer networks, JMLR.

Stochastic block multi-layer networks

- **Stochastic block model for multi-layer networks**[3]

$$Y_{uv}^{(k)}(t)|(s_u = a, s_v = b) \sim \text{Bern}\left(\pi_{ab}^{(k)}(t)\right)$$

$$\text{Logit}\left[\pi_{ab}^{(k)}(t)\right] = z(t) - d\left(x_a(t), x_b(t)\right) - d\left(x_a^{(k)}(t), x_b^{(k)}(t)\right),$$

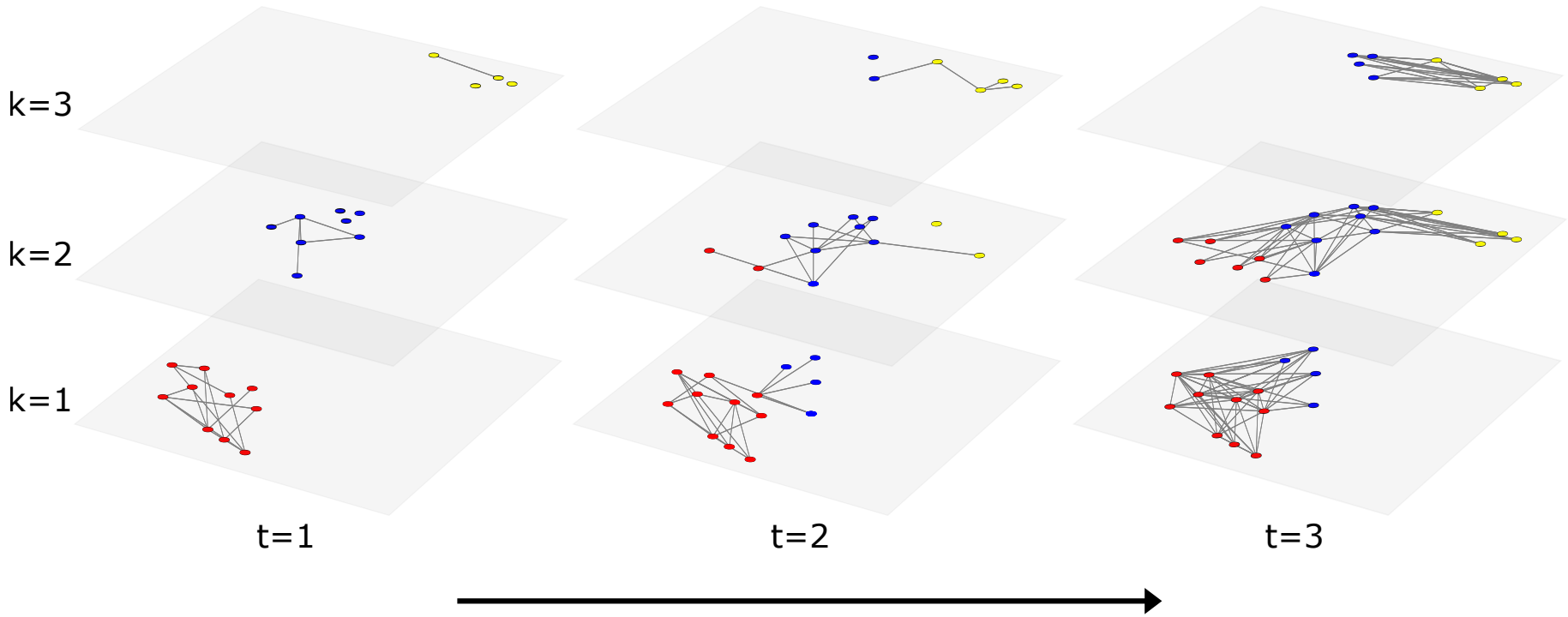
$$\text{Logit}\left[\pi_{aa}^{(k)}(t)\right] = z(t) + \beta x_a(t) + \gamma x_a^{(k)}(t),$$

$$s_u \sim \text{Categorical}(\omega_1, ..., \omega_B)$$

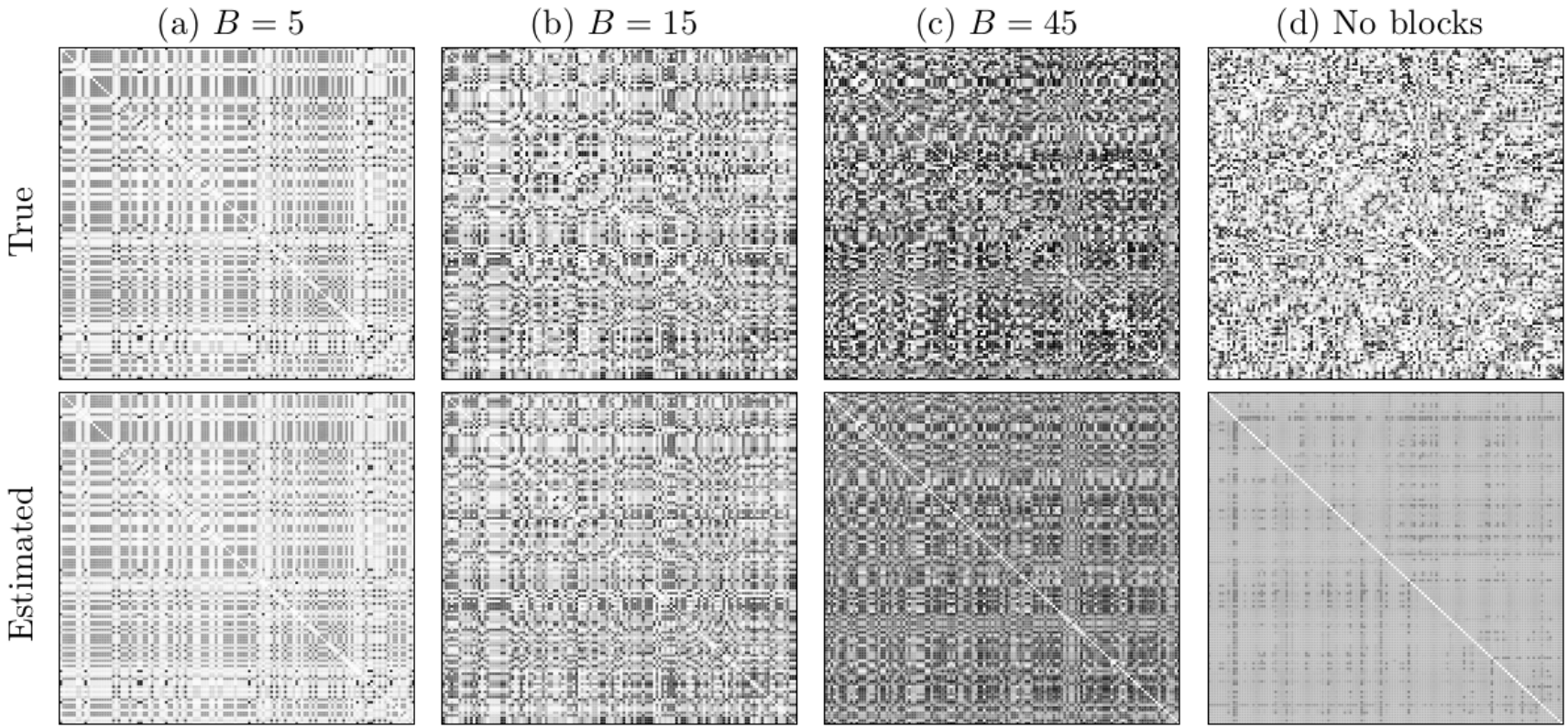
- Number of GPs: $1 + B + KB$ for B blocks. $B \ll N$.
- Need to learn the latent block indicators, s_u, for $u = 1, ..., N$.
- Gibbs sampling via Pólya-Gamma augmentation.

[3] Rodriguez-Deniz et (2022). A Multilayered Block Network Model to Forecast Large Dynamic Transportation Graphs: an Application to US Air Transport, *Transportation Research C*.

Stochastic block multi-layer networks

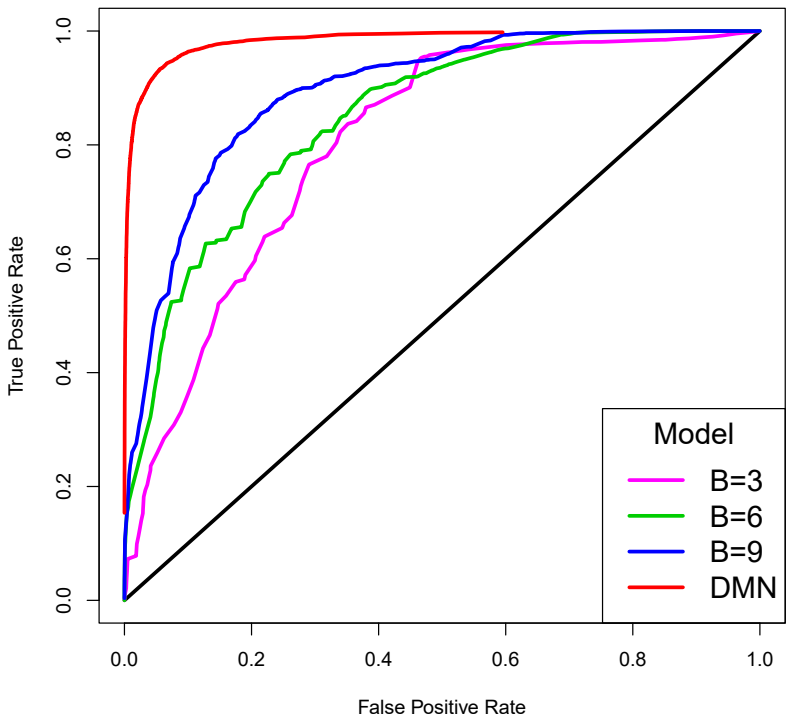


Fitting a $B = 10$ block multi-layer model

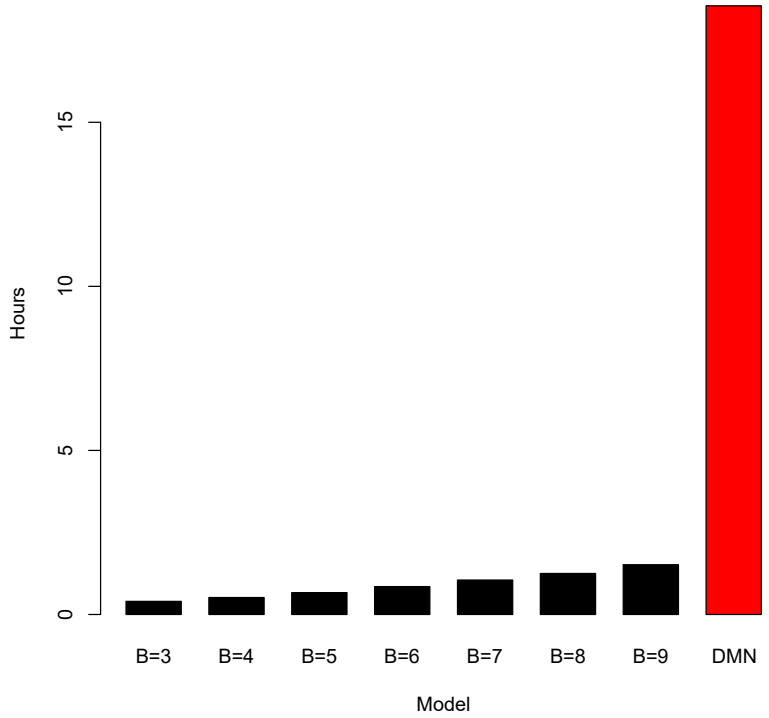


Predictive performance on 4 quarters test data

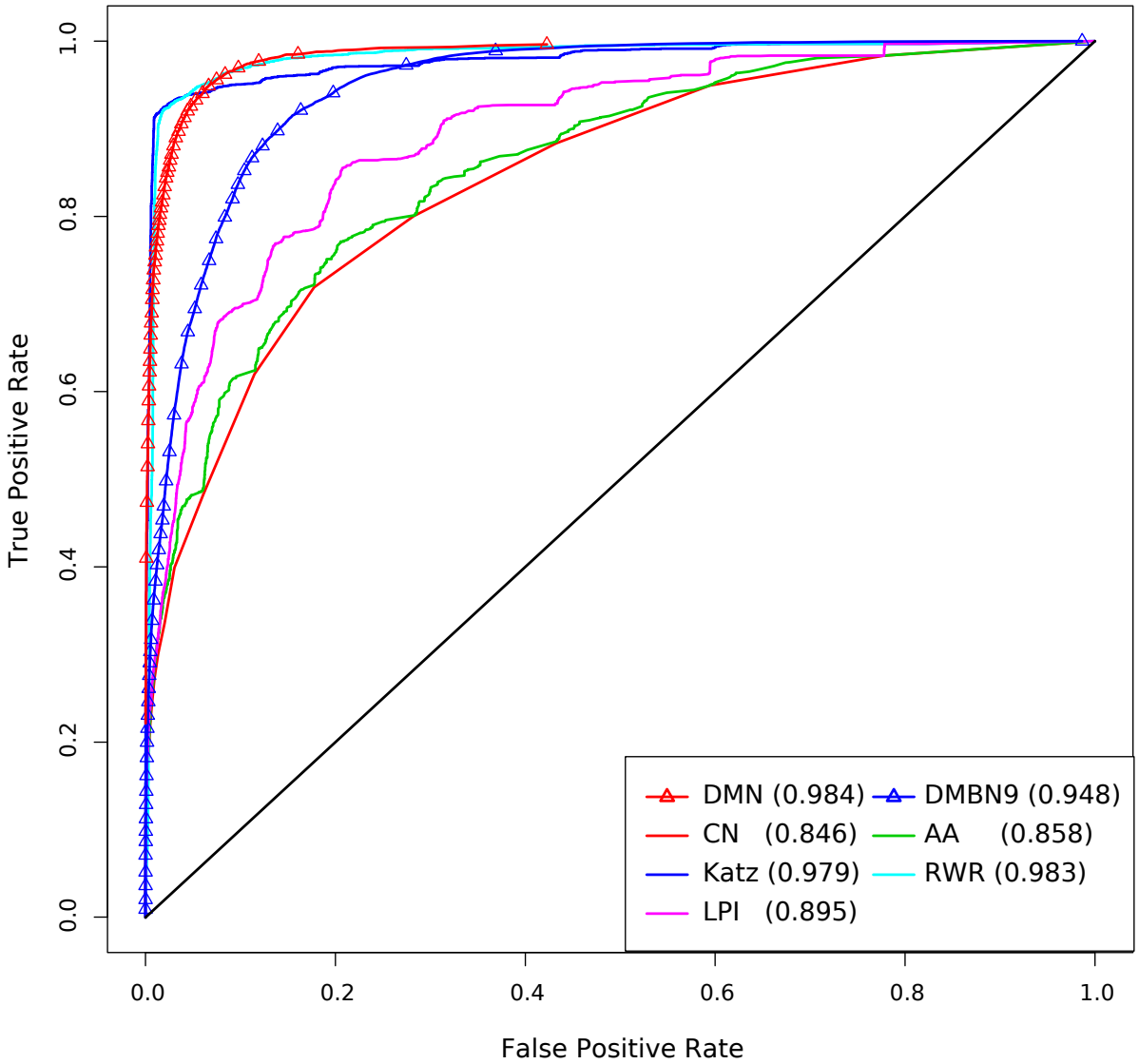
ROC curves



Computing times



Comparison to non-probabilistic SOTA



Bayesian optimization

- Aim: **maximization of expensive function**

$$\text{argmax}_{x \in \mathcal{X}} f(x)$$

- **Bayesian optimization**:
  - Assume $f \sim \mathcal{GP}$
  - Evaluate f at $x_1, x_2, ..., x_n$.
  - Update to posterior distribution $f|f(x_1), ..., f(x_n) \sim \mathcal{GP}$.
  - Use posterior of f to find a new x_{n+1}. **Explore** vs **Exploit**.
  - Iterate until convergence.
- Optimal x_{n+1} through an **acquisition function**.

Acquisition functions

- **Probability of Improvement (PI)**

$$a_{\mathrm{PI}}(\boldsymbol{x}) \equiv \Pr\left(f(\boldsymbol{x}) > f_{\mathrm{best}}\right),$$

  where f_{best} is the highest function value so far.

- **Expected Improvement (EI)** takes also into account the **size of the improvement**.

- Non-convex acquisition function optimization, but **deterministic** and **cheaper** than original problem.

- **Expected Improvement per Second** - takes a **known function cost** into account.

Marginal likelihood estimated from sampling

- **Marginal likelihood** $f(\boldsymbol{\theta}) \equiv \ln p(\boldsymbol{Y}_{1:T}|\boldsymbol{\theta})$ often estimated by **Monte Carlo sampling**.
- **Noisy** evaluations $\hat{f}(\boldsymbol{\theta})$.
- **Precision** of $\hat{f}(\boldsymbol{\theta})$ controlled via **number of samples** G.
- **Sampling efficiency** and therefore $\mathbb{V}(\hat{f}(\boldsymbol{\theta}))$ **varies over** $\boldsymbol{\theta}$**-space**, particularly when $\boldsymbol{\theta}$ contains prior/regularization hyperparameters.

Bayesian Optimization with Optimized Precision[4]

- BOOP:
  - **Early stopping of evaluation** when **PI**$< \alpha$.
  - G **random** - we don't know G until we visit $\boldsymbol{\theta}$.
  - **EI per second**, but with G **predicted** for every $\boldsymbol{\theta}$.
- Early stopping affects the **planning of future computations**.
- BOOP can try $\boldsymbol{\theta}$ with lower EI, if expected to be cheap.
- **Heteroscedastic GP regression model** for the estimates

$$\hat{f}(\boldsymbol{\theta}_i) = f(\boldsymbol{\theta}_i) + \epsilon_i, \qquad \epsilon_i \overset{\text{iid}}{\sim} N\left(0, \sigma^2(G_i)\right)$$

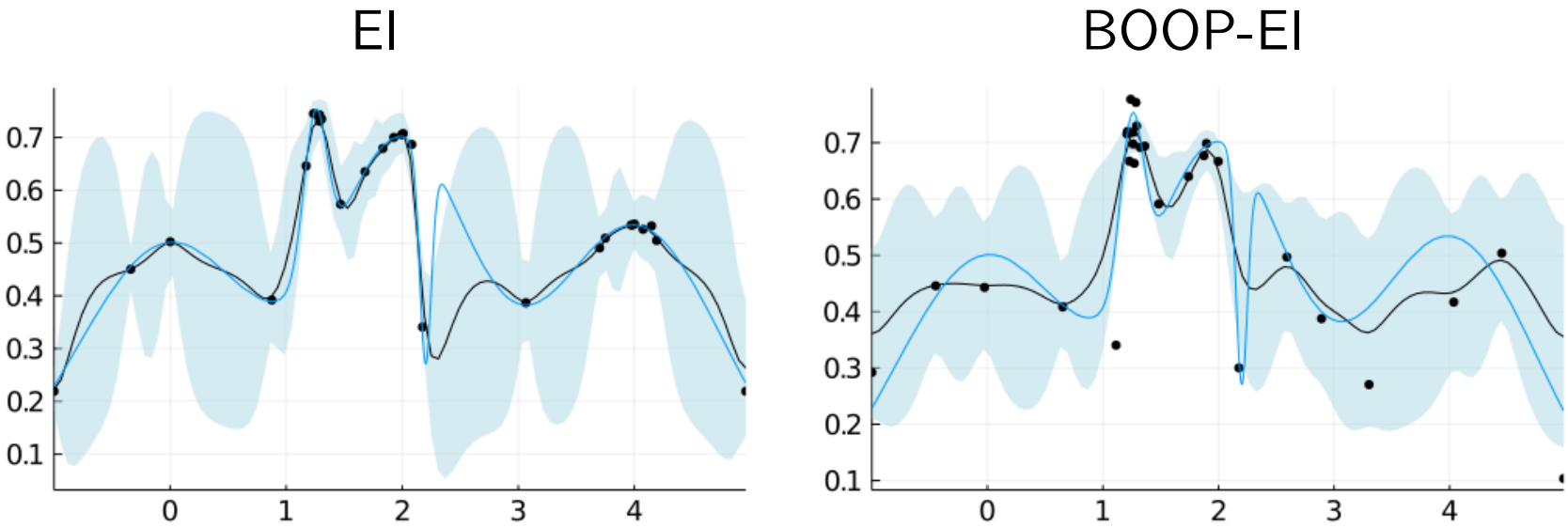
- GP for **predicting the number of samples** G:

$$\ln G_i = h(\boldsymbol{z}_i) + \varepsilon_i \qquad \varepsilon_i \overset{\text{iid}}{\sim} N\left(0, \psi^2\right),$$

where $\boldsymbol{z}$ are variables with predictive power for G, e.g. the hyperparameter values themselves, or $\hat{m}(\boldsymbol{\theta}) - f_{\max}$.

[4] Gustafsson et (2022). Bayesian Optimization of Hyperparameters when the Marginal Likelihood is Estimated by MCMC, *arXiv*.

BOOP in action - one example run



Bayesian VAR - BOOP finds estimates quicker

- 7 variable **steady-state BVAR** on US data.
- Gibbs sampling with **Chib's marginal likelihood estimator**.
- BO to find optimal prior hyperparameters:
  - Standard deviation first lag ($\lambda_1 \geq 0$)
  - Cross-equation shrinkage ($0 \leq \lambda_2 \leq 1$)
  - Lag length decay ($\lambda_3 \geq 0$)