Introduction aux Processus Gaussiens (1/2)

Application aux données temporelles

Clément Lejeune

company,
dept

4 novembre 2024

Contenu

Objectifs de cette présentation:

- Pédagogique (pas un état de l'art)

Contenu

Objectifs de cette présentation:

- Pédagogique (pas un état de l'art)
- Comprendre la philosophie (Bayésienne et non-paramétrique) des processus Gaussiens (GP)

Contenu

Objectifs de cette présentation:

- Pédagogique (pas un état de l'art)
- Comprendre la philosophie (Bayésienne et non-paramétrique) des processus Gaussiens (GP)
- Comprendre leurs interêts probabilistes/prédictifs

Contenu

Objectifs de cette présentation:

- Pédagogique (pas un état de l'art)
- Comprendre la philosophie (Bayésienne et non-paramétrique) des processus Gaussiens (GP)
- Comprendre leurs interêts probabilistes/prédictifs
- Comprendre les limites des GP "naïfs"

De quoi va-t-on parler ?

- Théorie: lois Gaussiennes, noyaux, probabilités conditionnelles

Contenu

Objectifs de cette présentation:

- Pédagogique (pas un état de l'art)
- Comprendre la philosophie (Bayésienne et non-paramétrique) des processus Gaussiens (GP)
- Comprendre leurs interêts probabilistes/prédictifs
- Comprendre les limites des GP "naïfs"

De quoi va-t-on parler ?

- Théorie: lois Gaussiennes, noyaux, probabilités conditionnelles
- Applicatif: forecasting, prédiction d'incertitude, complétion de valeurs manquantes, lissage

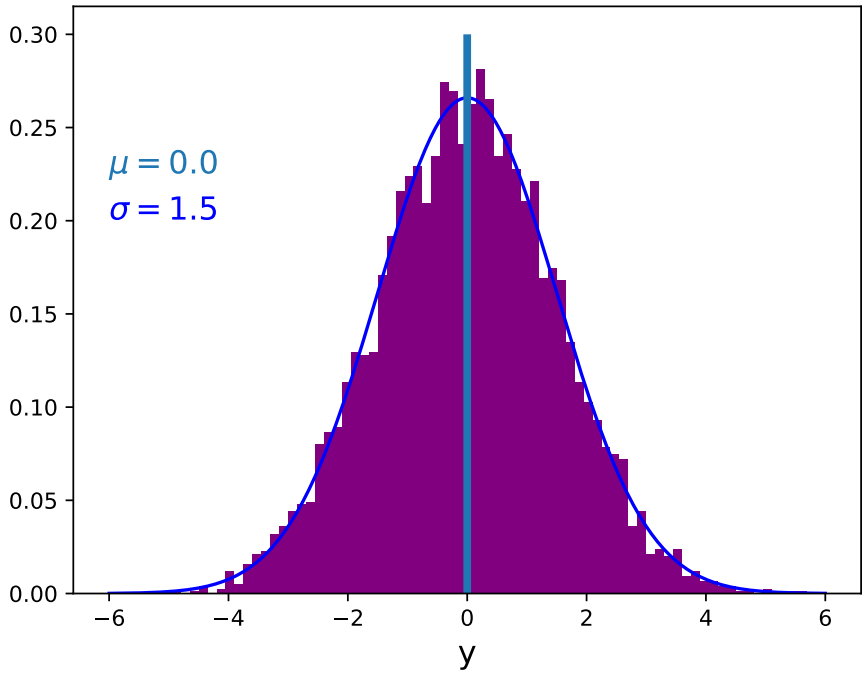
Gaussien: vecteur vs. processus

Construction d'un *GP*

Loi Gaussienne unidimensionnelle:

$$y \sim \mathcal{N}(m, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y-m)^2}{2\sigma^2}}$$

1. m: espérance (aka moyenne) de y
2. $\sigma > 0$: écart-type



Construction d'un *GP*

Loi Gaussienne multidimensionnelle (*vecteur* Gaussien): Distribution *jointe* des composantes d'un vecteur d-dimensionnel dont les *marginales* sont Gaussiennes (unidimensionnelles).

$$\mathbf{y} := [y_1, \ldots, y_d]^\top \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d \det |\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{y}-\boldsymbol{\mu})}$$

1. $\boldsymbol{\mu} \in \mathbb{R}^d$: *vecteur moyen* $\implies$ μ_j: moyenne de la Gaussienne y_j
2. $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$ définie positive[1]: *matrice de covariance*

[1] i.e. $\mathbf{a}^\top \boldsymbol{\Sigma} \mathbf{a} > 0$ (donc symmétrique)

Construction d'un *GP*

Loi Gaussienne multidimensionnelle (*vecteur* Gaussien): Distribution *jointe* des composantes d'un vecteur d-dimensionnel dont les *marginales* sont Gaussiennes (unidimensionnelles).

$$\mathbf{y} := [y_1, \ldots, y_d]^\top \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d \det|\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{y}-\boldsymbol{\mu})}$$

1. $\boldsymbol{\mu} \in \mathbb{R}^d$: *vecteur moyen* $\Longrightarrow$ μ_j: moyenne de la Gaussienne y_j
2. $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$ définie positive[1]: *matrice de covariance*

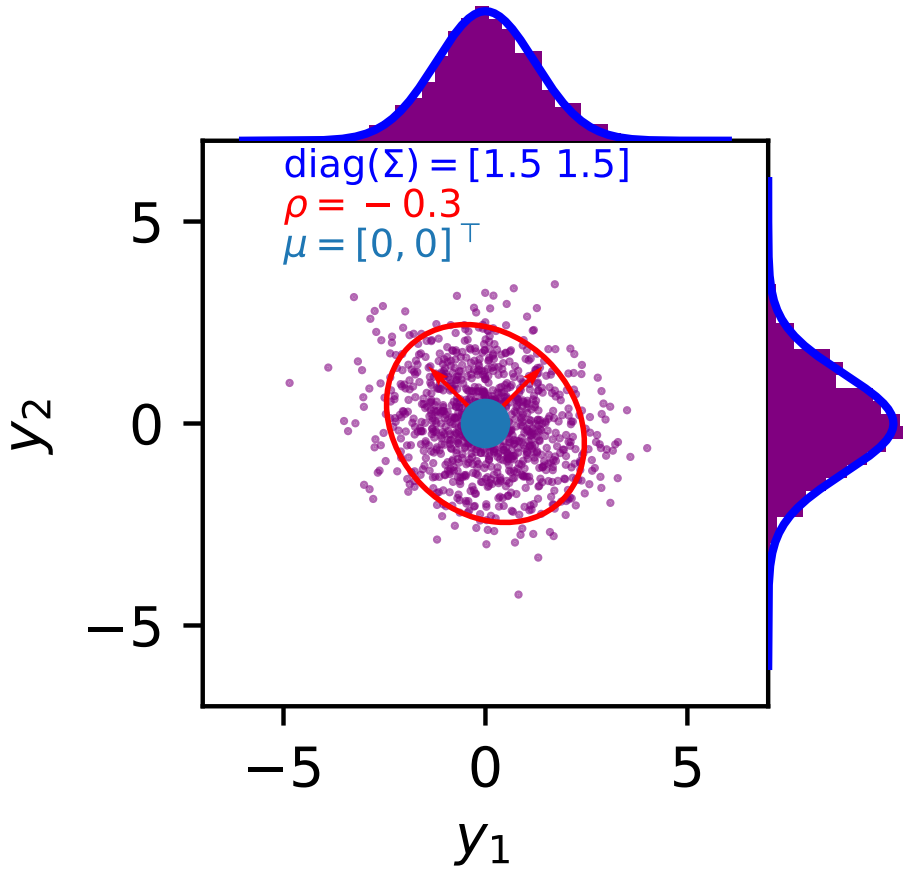
$$\boldsymbol{\Sigma} = \begin{pmatrix} \Sigma_1^2 & \Sigma_{12} & \cdots & \Sigma_{1d} \\ \Sigma_{21} & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{d1} & \cdots & \cdots & \Sigma_d^2 \end{pmatrix} \Longrightarrow \begin{cases} \Sigma_j : \text{écart-type de } y_j \\ \Sigma_{ij} : \text{covariance entre les marginales } y_i \text{ et } y_j \end{cases}$$

[1] i.e. $\mathbf{a}^\top \boldsymbol{\Sigma} \mathbf{a} > 0$ (donc symétrique)

Construction d'un *GP*

Cas $d = 2$:

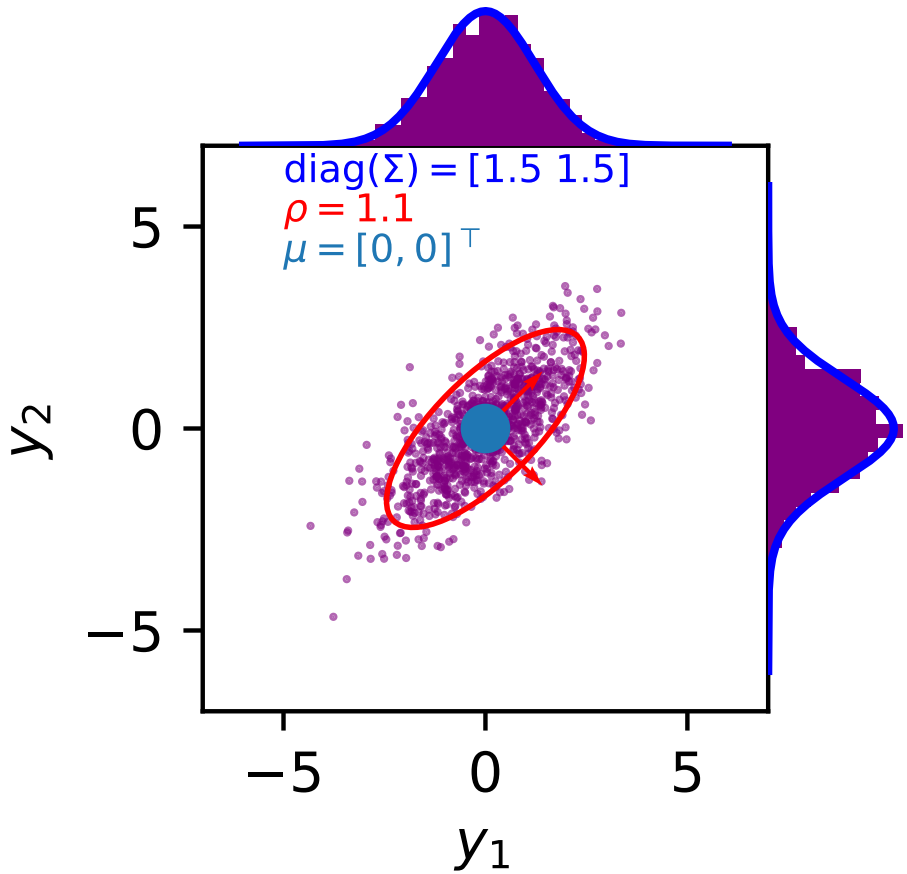
$$\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \Sigma_1 & \Sigma_{12} \\ \Sigma_{21} & \Sigma_2 \end{pmatrix} = \begin{pmatrix} 1.5 & \rho = -0.3 \\ \rho = -0.3 & 1.5 \end{pmatrix}$$



Construction d'un *GP*

Cas $d = 2$:

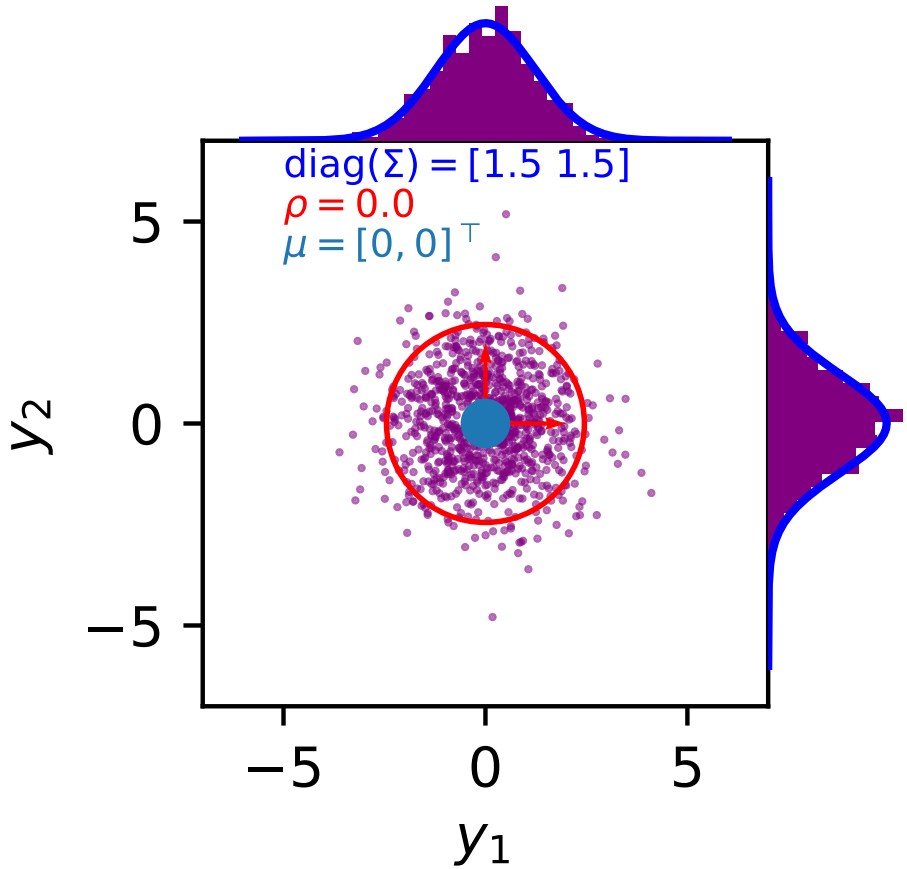
$$\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \Sigma_1 & \Sigma_{12} \\ \Sigma_{21} & \Sigma_2 \end{pmatrix} = \begin{pmatrix} 1.5 & \rho = 1.1 \\ \rho = 1.1 & 1.5 \end{pmatrix}$$



Construction d'un *GP*

Cas $d = 2$:

$$\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \Sigma_1 & \Sigma_{12} \\ \Sigma_{21} & \Sigma_2 \end{pmatrix} = \begin{pmatrix} 1.5 & \rho = 0 \\ \rho = 0 & 1.5 \end{pmatrix}$$



Construction d'un *GP*

Cas $d = 2$:

$$\boldsymbol{\mu} = \begin{pmatrix} 0 \\ 0 \\ \dots \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} 1.5 & 0.99 & 0.98 & 0.96 & 0.94 \\ 0.99 & 1.5 & 0.99 & 0.98 & 0.96 \\ 0.98 & 0.99 & 1.5 & 0.99 & 0.98 \\ 0.96 & 0.98 & 0.99 & 1.5 & 0.99 \\ 0.94 & 0.96 & 0.98 & 0.99 & 1.5 \end{pmatrix}$$

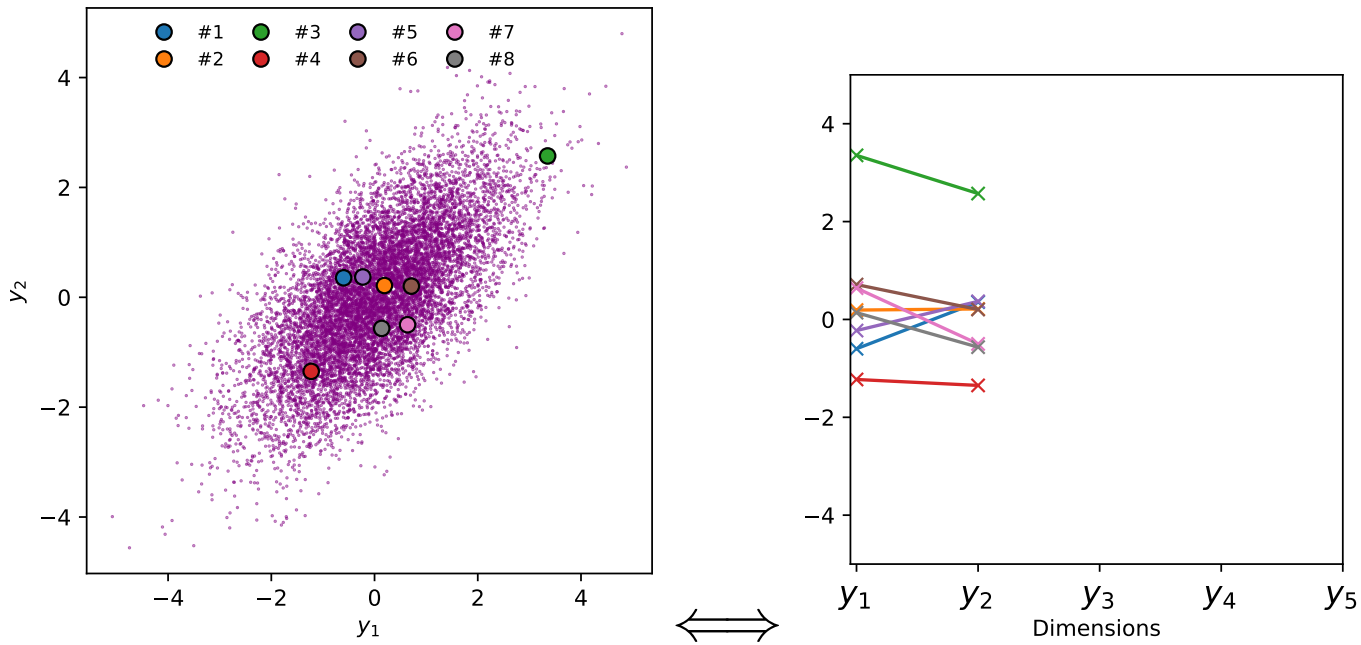


Figure: Dimensions $j = 1, 2$. Gauche: 10^4 +8 échantillons. Droite: Réindéxation des 8 *mêmes* échantillons.

Construction d'un *GP*

Cas $d = 5$:

$$\boldsymbol{\mu} = \begin{pmatrix} 0 \\ 0 \\ \dots \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} 1.5 & 0.99 & 0.98 & 0.96 & 0.94 \\ 0.99 & 1.5 & 0.99 & 0.98 & 0.96 \\ 0.98 & 0.99 & 1.5 & 0.99 & 0.98 \\ 0.96 & 0.98 & 0.99 & 1.5 & 0.99 \\ 0.94 & 0.96 & 0.98 & 0.99 & 1.5 \end{pmatrix}$$

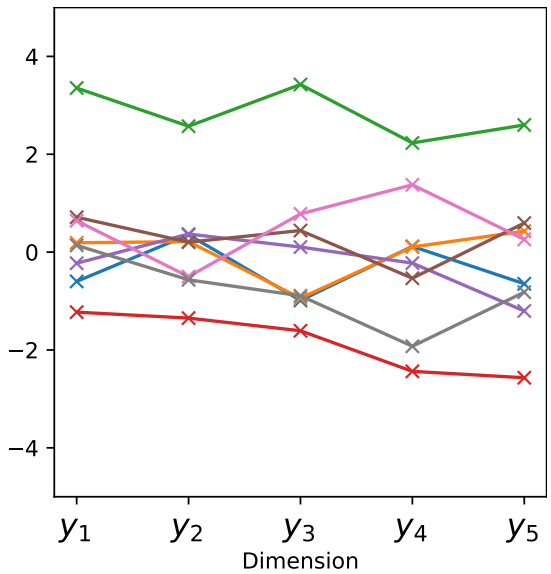


Figure: Réindéxation de 8 échantillons, dimensions $j = 1, \dots, 5$. Chaque "courbe" est un tirage de $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

Construction d'un *GP*

Cas $d = 50$:

$$\boldsymbol{\mu} = \begin{pmatrix} 0 \\ 0 \\ \dots \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} 1.5 & 0.998 & 0.993 & 0.986 & 0.975 & 0.962 & 0.946 & 0.927 & \cdots \\ 0.998 & 1.5 & 0.998 & 0.993 & 0.986 & 0.975 & 0.962 & 0.946 & \cdots \\ 0.993 & 0.998 & 1.5 & 0.998 & 0.993 & 0.986 & 0.975 & 0.962 & \cdots \\ 0.986 & 0.993 & 0.998 & 1.5 & 0.998 & 0.993 & 0.986 & 0.975 & \cdots \\ 0.975 & 0.986 & 0.993 & 0.998 & 1.5 & 0.998 & 0.993 & 0.986 & \cdots \\ 0.962 & 0.975 & 0.986 & 0.993 & 0.998 & 1.5 & 0.998 & 0.993 & \cdots \\ 0.946 & 0.962 & 0.975 & 0.986 & 0.993 & 0.998 & 1.5 & 0.998 & \cdots \\ 0.927 & 0.946 & 0.962 & 0.975 & 0.986 & 0.993 & 0.998 & 1.5 & \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 1.5 \end{pmatrix}$$

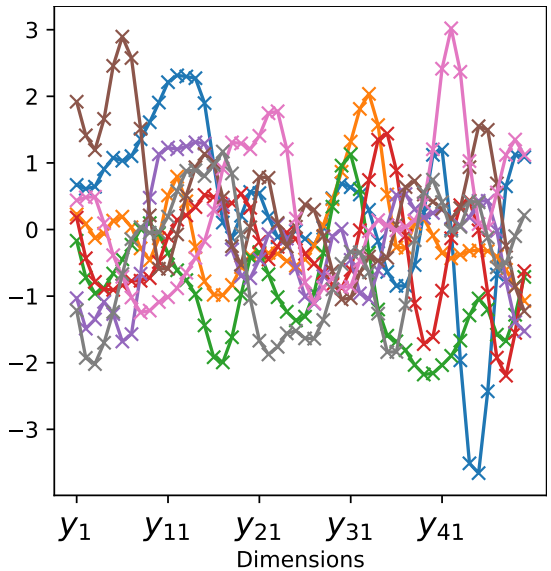


Figure: Réindéxation de 8 échantillons, dimensions $j = 1, \dots, 50$

Construction d'un *GP*

Cas $d = \infty$: chaque courbe est une *collection infinie* de valeurs, que l'on peut voir comme une fonction:

1. $\boldsymbol{\mu}$: vecteur de taille infinie $\Leftrightarrow$ fonction moyenne $\mathbb{E}(y(x)) = m(\mathbf{x})$
2. $\boldsymbol{\Sigma}$: matrice de taille infinie $\Leftrightarrow$ noyau de covariance ou "*kernel*" $\mathbb{C}(y(\mathbf{x}), y(\mathbf{x'})) = k(\mathbf{x}, \mathbf{x'})$

Construction d'un *GP*

Cas $d = \infty$: chaque courbe est une *collection infinie* de valeurs, que l'on peut voir comme une fonction:

1. $\boldsymbol{\mu}$: vecteur de taille infinie $\Leftrightarrow$ fonction moyenne $\mathbb{E}(y(x)) = m(\mathbf{x})$
2. $\boldsymbol{\Sigma}$: matrice de taille infinie $\Leftrightarrow$ noyau de covariance ou "*kernel*" $\mathbb{C}(y(\mathbf{x}), y(\mathbf{x'})) = k(\mathbf{x}, \mathbf{x'})$

Processus Gaussien $y(\mathbf{x}) \sim GP(m, k)$ Rasmussen and Williams 2006

Un processus Gaussien (GP) est une distribution de probabilités sur un espace de *fonctions*, $y(\mathbf{x}) : \mathcal{X} \to \mathcal{Y}$, telle que toute collection finie $[y(x_i), \ldots, y(x_j), \ldots y(x_k)], \forall i, j, k$ forme un vecteur Gaussien.

Construction d'un *GP*

- *GP* peut être vu comme une réindéxation d'un vecteur "infini"

Construction d'un *GP*

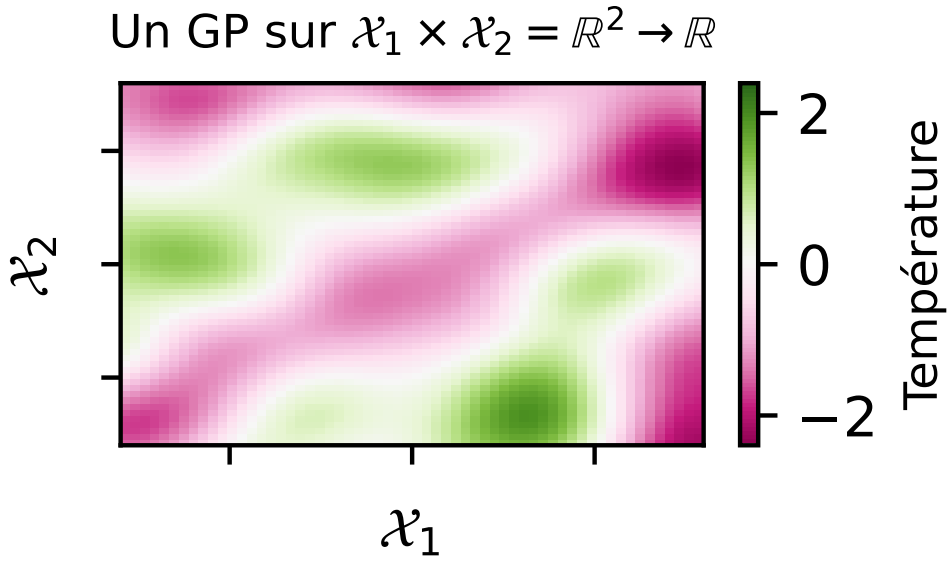
- *GP* peut être vu comme une réindéxation d'un vecteur "infini"
- Cas temporal: $y(t) : \mathbb{R}^+ \to \mathbb{R}$, *n'importe qu'elle discrétisation* de y doit former un vecteur Gaussien, en prenant par ex 2 entrées de y:

$$y(\mathbf{x}) \sim GP(m, k) \implies y([t_i, t_j]) \sim \mathcal{N}(\begin{pmatrix} \mu_i \\ \mu_j \end{pmatrix}, \begin{pmatrix} \Sigma_i^2 & \Sigma_{ij} \\ \Sigma_{ij} & \Sigma_j^2 \end{pmatrix})$$

- Exemple visu: cf. exemples introductifs

Construction d'un *GP*

- Cas spatial: $y(\mathbf{x}) : \mathbb{R}^2 \to \mathcal{Y} = \mathbb{R}$ (ex: température en 2D)
- Visu d'un échantillon:



Construction d'un *GP*

- On impose souvent une forme simple sur $\mathbb{E}(y(x)) = m(x)$ comme: $\boldsymbol{\beta}^\top \mathbf{x}$, $\mathbf{0}$

Construction d'un *GP*

- On impose souvent une forme simple sur $\mathbb{E}(y(x)) = m(x)$ comme: $\boldsymbol{\beta}^\top \mathbf{x}$, $\mathbf{0}$
- $y : \mathcal{X} \to \mathcal{Y}$ est une *fonction*: il suffit de bien choisir $\mathcal{X}$ et $\mathcal{Y}$
  - $\mathcal{X} = \mathbb{R}^+$ et $\mathcal{Y} = \mathbb{R} \implies y =$ *série temporelle* Rasmussen **and** Williams 2006

Construction d'un *GP*

- On impose souvent une forme simple sur $\mathbb{E}(y(x)) = m(x)$ comme: $\boldsymbol{\beta}^\top \mathbf{x}$, $\mathbf{0}$
- $y : \mathcal{X} \to \mathcal{Y}$ est une *fonction*: il suffit de bien choisir $\mathcal{X}$ et $\mathcal{Y}$
  - $\mathcal{X} = \mathbb{R}^+$ et $\mathcal{Y} = \mathbb{R} \implies y =$ *série temporelle* Rasmussen **and** Williams 2006
  - $\mathcal{X} = [a, b] \times [c, d]$ (ou $\mathbb{R}^2$) et $\mathcal{Y} = \mathbb{R} \implies y =$ *champ spatial* (météo, hydro, etc)

Construction d'un *GP*

- On impose souvent une forme simple sur $\mathbb{E}(y(x)) = m(x)$ comme: $\beta^\top \mathbf{x}$, $\mathbf{0}$
- $y : \mathcal{X} \to \mathcal{Y}$ est une *fonction*: il suffit de bien choisir $\mathcal{X}$ et $\mathcal{Y}$
  - $\mathcal{X} = \mathbb{R}^+$ et $\mathcal{Y} = \mathbb{R} \implies y =$ *série temporelle* Rasmussen **and** Williams 2006
  - $\mathcal{X} = [a, b] \times [c, d]$ (ou $\mathbb{R}^2$) et $\mathcal{Y} = \mathbb{R} \implies y =$ *champ spatial* (météo, hydro, etc)
  - $\mathcal{X} = [a, b] \times [c, d] \times \mathbb{R}^+$ et $\mathcal{Y} = \mathbb{R} \implies y =$ *série temporelle d'images*

Construction d'un *GP*

- On impose souvent une forme simple sur $\mathbb{E}(y(x)) = m(x)$ comme: $\boldsymbol{\beta}^\top \mathbf{x}$, $\mathbf{0}$
- $y : \mathcal{X} \to \mathcal{Y}$ est une *fonction*: il suffit de bien choisir $\mathcal{X}$ et $\mathcal{Y}$
  - $\mathcal{X} = \mathbb{R}^+$ et $\mathcal{Y} = \mathbb{R} \implies y =$ *série temporelle* Rasmussen **and** Williams 2006
  - $\mathcal{X} = [a, b] \times [c, d]$ (ou $\mathbb{R}^2$) et $\mathcal{Y} = \mathbb{R} \implies y =$ *champ spatial* (météo, hydro, etc)
  - $\mathcal{X} = [a, b] \times [c, d] \times \mathbb{R}^+$ et $\mathcal{Y} = \mathbb{R} \implies y =$ *série temporelle d'images*
  - $\mathcal{X} = \mathbb{R}^d$ et $\mathcal{Y} = \{1 \dots K\} \implies y =$ *classificateur* d'images/pixels (ex: segmentation) Hensman **andothers** 2015
  - $\mathcal{X}$ et $\mathcal{Y}$ peuvent être des espaces de graphes Borovitskiy **andothers** 2021
  - . . . etc, etc, etc.

Gaussien: vecteur vs. processus

Le kernel

- $k(\mathbf{x}, \mathbf{x}') = \mathbb{C}(y(\mathbf{x}), y(\mathbf{x}'))$ représente la *similarité de y* entre deux entrées $\mathbf{x}$ et $\mathbf{x}'$, donc doit générer une matrice de covariance valide[2]

[2] $\sum_{i,j}^{n} a_i a_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0, \forall \mathbf{a} = [a_1, \ldots, a_n]$

[3] C-à-d être un noyau reproduisant d'un espace de Hilbert: $y(x_0) = \sum_{i=1}^{\infty} y(x_i) k(x_i, x_0)$

Le kernel

- $k(\mathbf{x}, \mathbf{x}') = \mathbb{C}(y(\mathbf{x}), y(\mathbf{x}'))$ représente la *similarité de y* entre deux entrées $\mathbf{x}$ et $\mathbf{x}'$, donc doit générer une matrice de covariance valide[2]
- k défini *implicitement* les propriétés fonctionnelles de y[3]

[2] $\sum_{i,j}^{n} a_i a_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0, \forall \mathbf{a} = [a_1, \ldots, a_n]$

[3] C-à-d être un noyau reproduisant d'un espace de Hilbert: $y(x_0) = \sum_{i=1}^{\infty} y(x_i) k(x_i, x_0)$

Le kernel

- $k(\mathbf{x}, \mathbf{x}') = \mathbb{C}(y(\mathbf{x}), y(\mathbf{x}'))$ représente la *similarité de y* entre deux entrées $\mathbf{x}$ et $\mathbf{x}'$, donc doit générer une matrice de covariance valide[2]
- k défini *implicitement* les propriétés fonctionnelles de y[3]
- Kernel: fonction qui dépend aussi *d'hyperparamètres* $\boldsymbol{\theta}$

[2] $\sum_{i,j}^{n} a_i a_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0, \forall \mathbf{a} = [a_1, \ldots, a_n]$

[3] C-à-d être un noyau reproduisant d'un espace de Hilbert: $y(x_0) = \sum_{i=1}^{\infty} y(x_i) k(x_i, x_0)$

Le kernel

- $k(\mathbf{x}, \mathbf{x}') = \mathbb{C}(y(\mathbf{x}), y(\mathbf{x}'))$ représente la *similarité de y* entre deux entrées $\mathbf{x}$ et $\mathbf{x}'$, donc doit générer une matrice de covariance valide[2]
- k défini *implicitement* les propriétés fonctionnelles de y[3]
- Kernel: fonction qui dépend aussi *d'hyperparamètres* $\boldsymbol{\theta}$
- Et donc le nerf d'un *GP* est essentiellement dans son kernel (noyau)

[2] $\sum_{i,j}^{n} a_i a_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0, \forall \mathbf{a} = [a_1, \ldots, a_n]$

[3] C-à-d être un noyau reproduisant d'un espace de Hilbert: $y(x_0) = \sum_{i=1}^{\infty} y(x_i) k(x_i, x_0)$

Le kernel

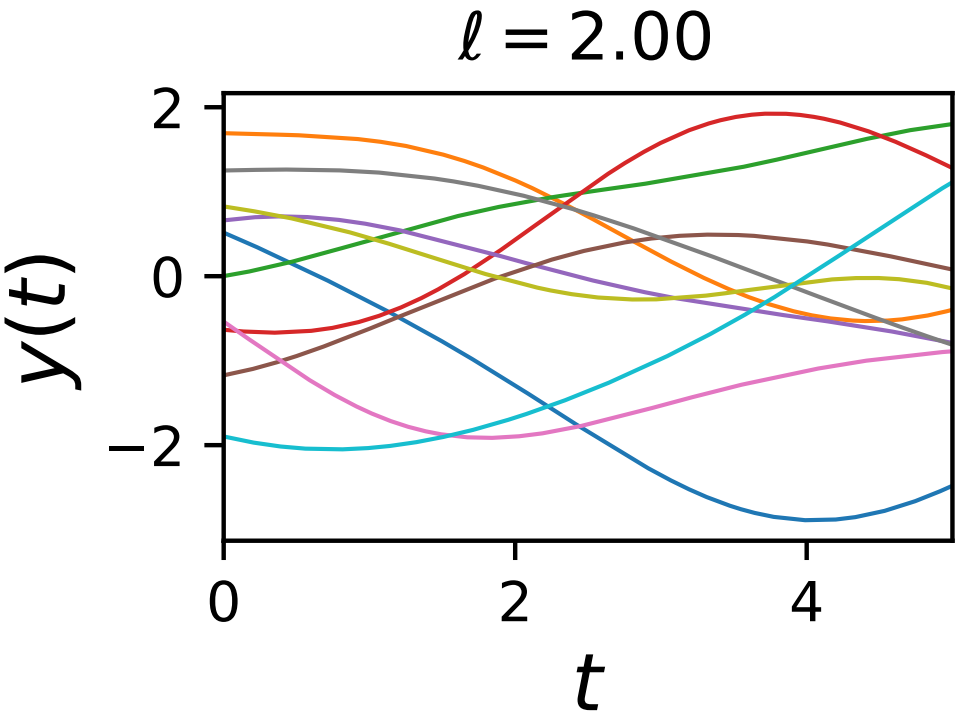
- Quelques exemples de $GP(0, k)$ défini sur $\mathcal{X} = \mathbb{R}^+$ et $\mathcal{Y} = \mathbb{R}$ (série temporelle)
- kernel de série temporelle $\leftrightarrow$ fonction d'*autocovariance*
- Visualiser: bonne manière de comprendre un kernel

Le kernel

- Exponentiel quadratique (aka "Radial basis", "Gaussian", "Heat" kernel): $k(t, t') = e^{-\frac{(t-t')^2}{2\ell^2}}$
- $GP(0, k)$ génère des y infiniment différentiables par rapport à t
- ℓ contrôle le lissé (ici au sens oscillatoir) des générées

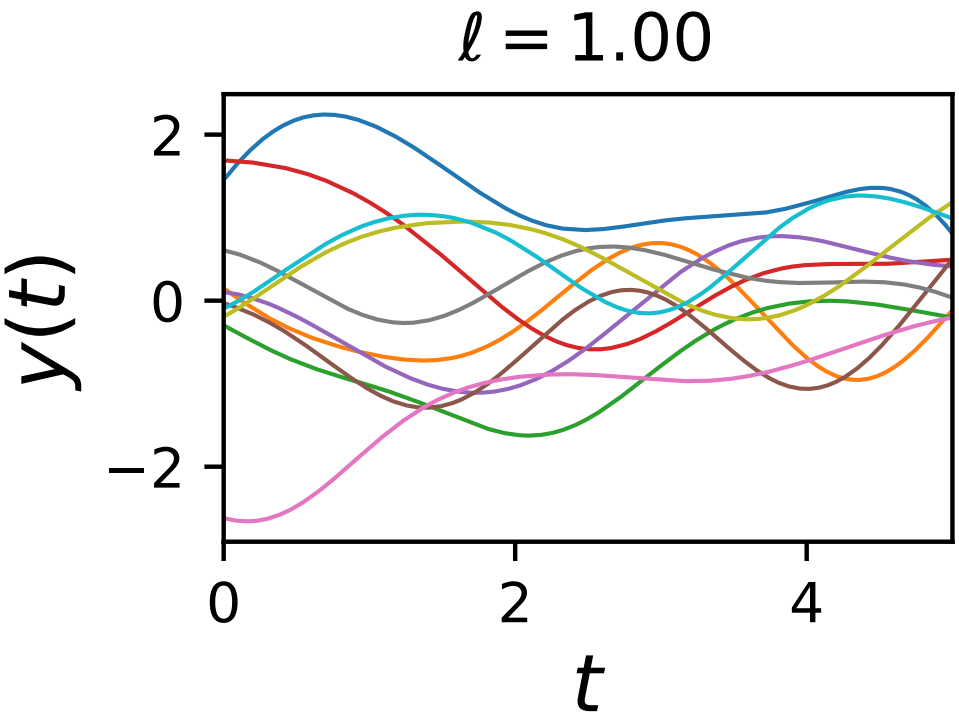
Le kernel

- Exponentiel quadratique (aka "Radial basis", "Gaussian", "Heat" kernel): $k(t, t') = e^{-\frac{(t-t')^2}{2\ell^2}}$
- $GP(0, k)$ génère des y infiniment différentiables par rapport à t
- ℓ contrôle le lissé (ici au sens oscillatoir) des générées



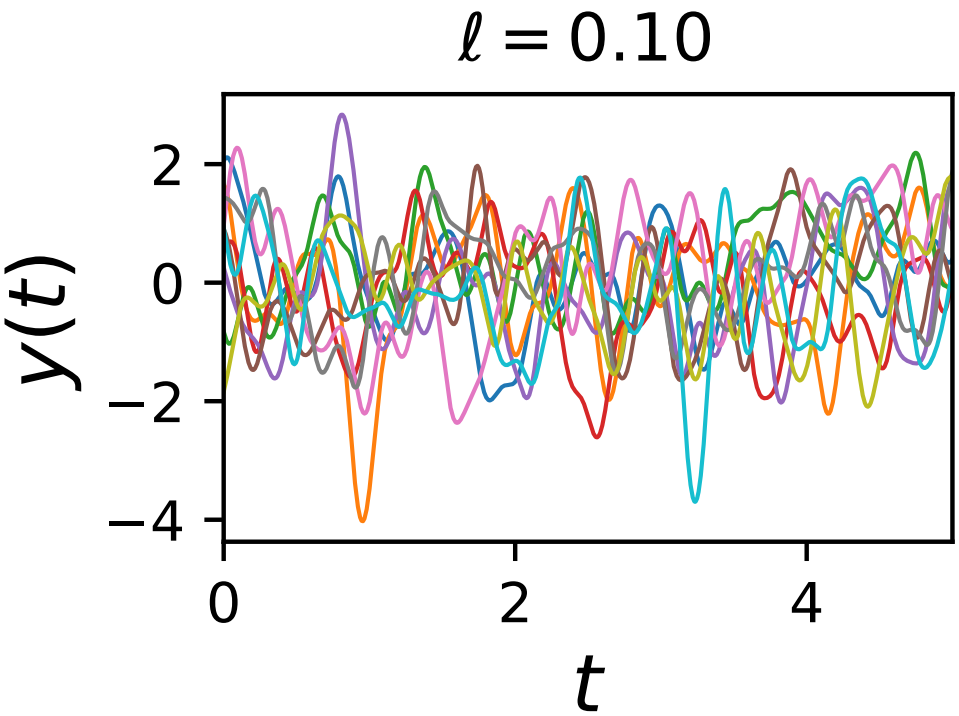
Le kernel

- Exponentiel quadratique (aka "Radial basis", "Gaussian", "Heat" kernel): $k(t, t') = e^{-\frac{(t-t')^2}{2\ell^2}}$
- $GP(0, k)$ génère des y infiniment différentiables par rapport à t
- ℓ contrôle le lissé (ici au sens oscillatoir) des générées



Le kernel

- Exponentiel quadratique (aka "Radial basis", "Gaussian", "Heat" kernel): $k(t, t') = e^{-\frac{(t-t')^2}{2\ell^2}}$
- $GP(0, k)$ génère des y infiniment différentiables par rapport à t
- ℓ contrôle le lissé (ici au sens oscillatoir) des générées

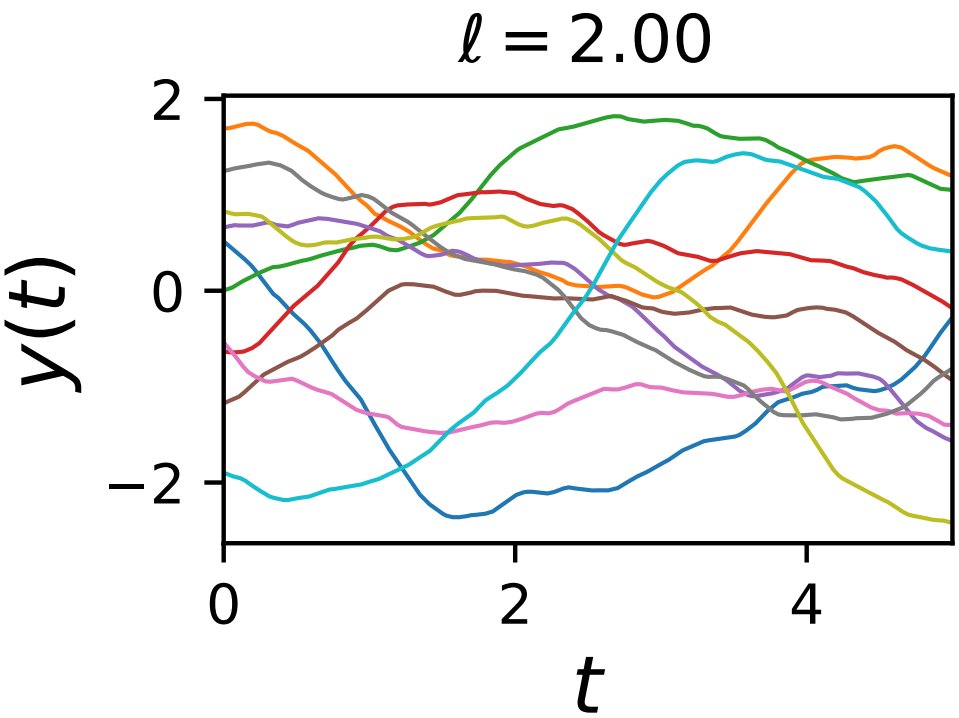


Le kernel

- kernel de Matérn: $k(t, t') = (1 + \sqrt{\frac{3|t-t'|}{\ell}})e^{-\sqrt{\frac{3|t-t'|}{\ell}}}$
- Génère des y différentiables une seule fois par rapport à t
- ℓ: même interprétation que le kernel exponentiel quadratique

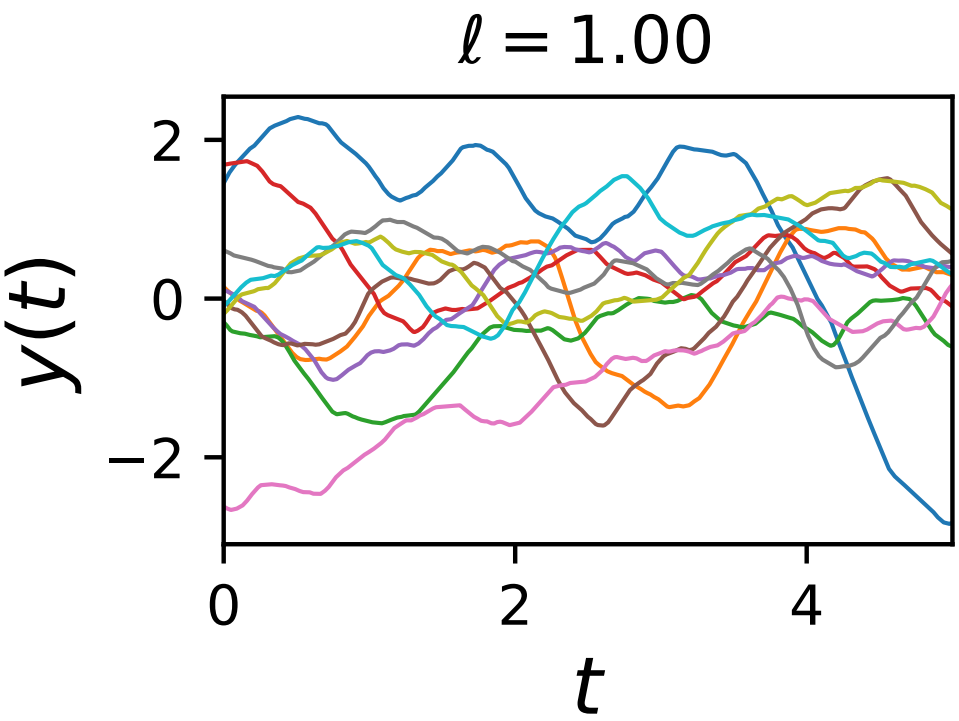
Le kernel

- kernel de Matérn: $k(t, t') = (1 + \sqrt{\frac{3|t-t'|}{\ell}})e^{-\sqrt{\frac{3|t-t'|}{\ell}}}$
- Génère des y différentiables une seule fois par rapport à t
- ℓ: même interprétation que le kernel exponentiel quadratique



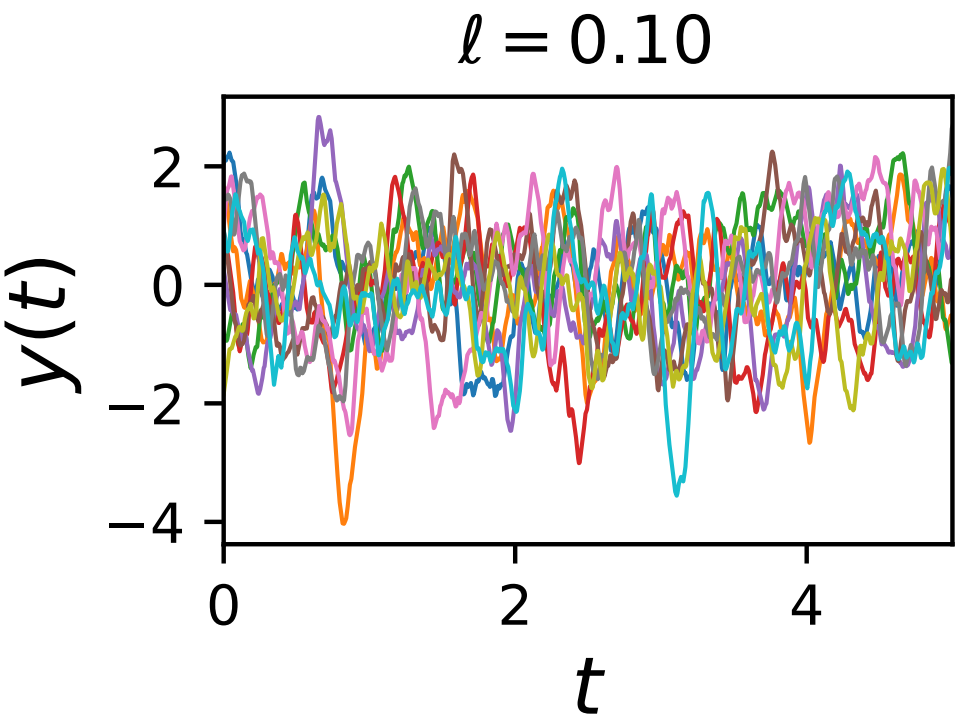
Le kernel

- kernel de Matérn: $k(t, t') = (1 + \sqrt{\frac{3|t-t'|}{\ell}})e^{-\sqrt{\frac{3|t-t'|}{\ell}}}$
- Génère des y différentiables une seule fois par rapport à t
- ℓ: même interprétation que le kernel exponentiel quadratique



Le kernel

- kernel de Matérn: $k(t, t') = (1 + \sqrt{\frac{3|t-t'|}{\ell}})e^{-\sqrt{\frac{3|t-t'|}{\ell}}}$
- Génère des y différentiables une seule fois par rapport à t
- ℓ: même interprétation que le kernel exponentiel quadratique

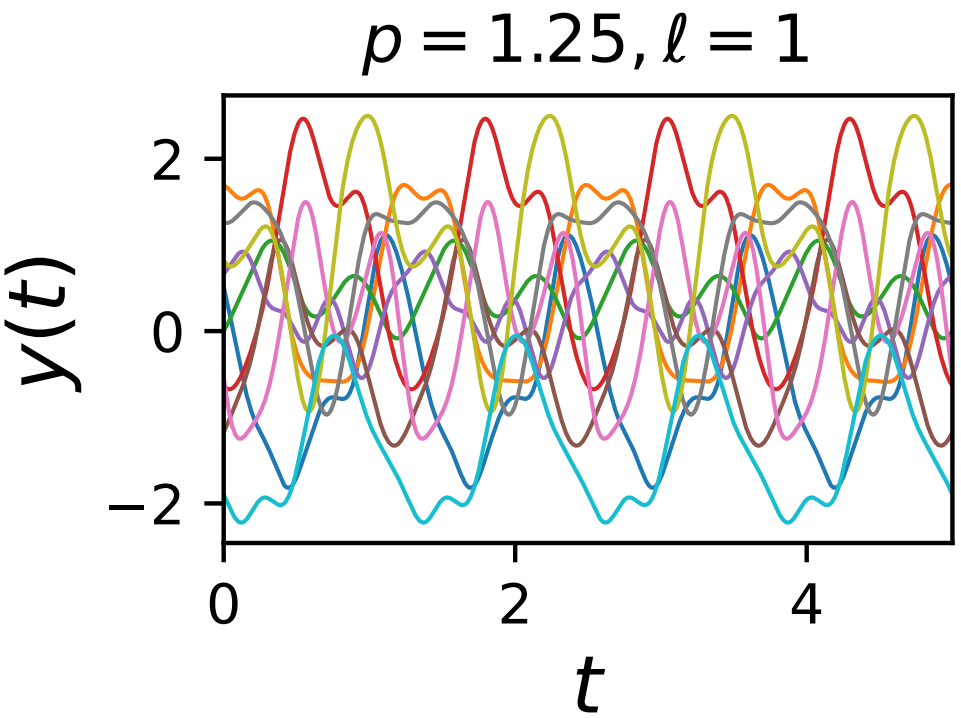


Le kernel

- kernel locallement-périodique: $k(t, t') = e^{-\frac{2\sin^2(\pi(t-t')/T_0)}{\ell^2}}$
- Génère des y périodiques ET lisses

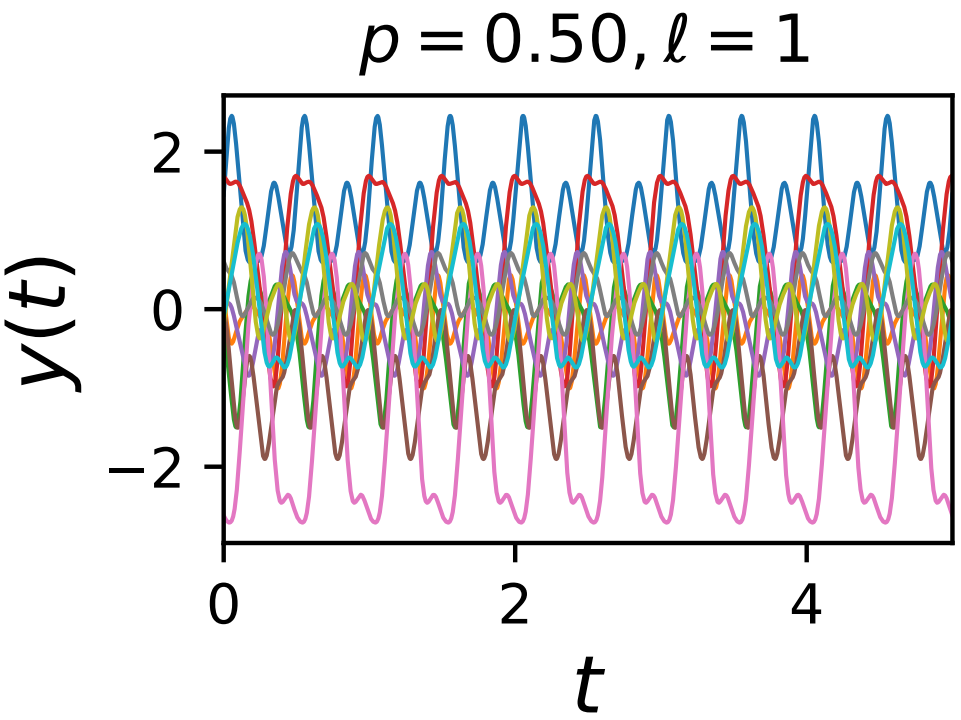
Le kernel

- kernel locallement-périodique: $k(t, t') = e^{-\frac{2\sin^2(\pi(t-t')/T_0)}{\ell^2}}$
- Génère des y périodiques ET lisses



Le kernel

- kernel locallement-périodique: $k(t, t') = e^{-\frac{2\sin^2(\pi(t-t')/T_0)}{\ell^2}}$
- Génère des y périodiques ET lisses

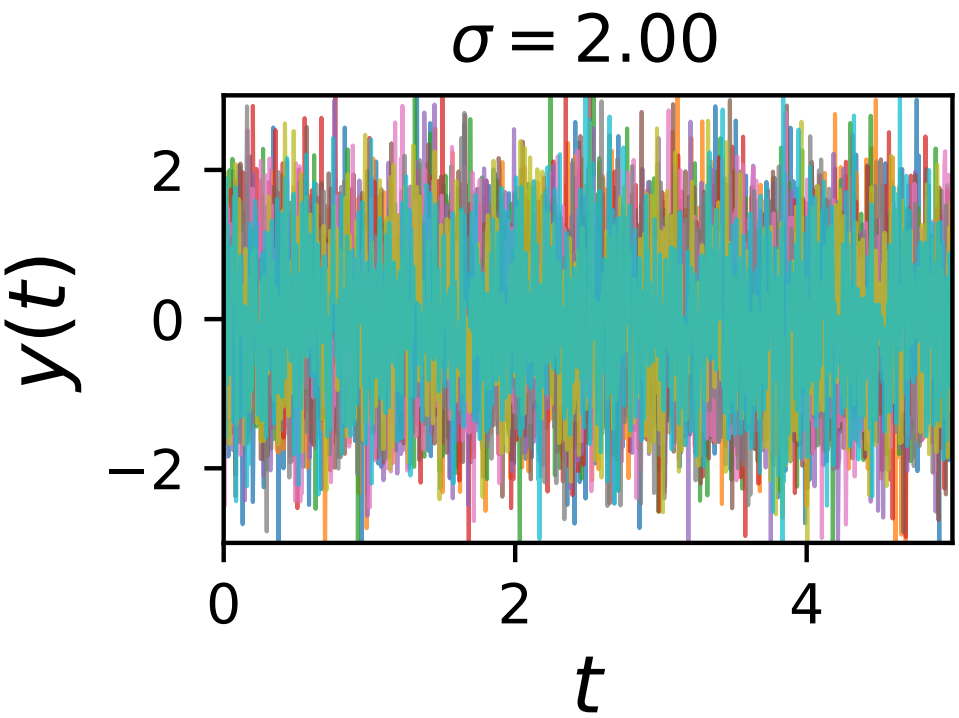


Le kernel

- kernel de bruit blanc:

$$k(t, t') = \begin{cases} \sigma^2 \text{ si } t = t' \\ 0 \text{ sinon} \end{cases}$$

- Aucune corrélation. σ contrôle l'amplitude (variance) du bruit



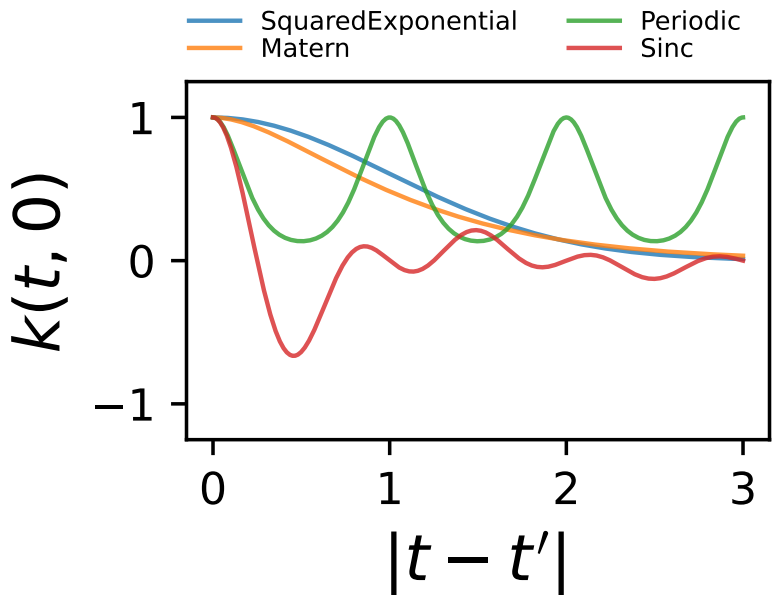
Le kernel

- Aussi, des kernels dans le domaine fréquentiel (Fourier)Parra **and** Tobar 2017[4]

[4] ex: $\sum_{q=0}^{Q} \sigma_q^2 e^{-2\pi^2 \Sigma_q (t-t')^2} \cos(2\pi \mu_q (t-t'))$

Le kernel

- Aussi, des kernels dans le domaine fréquentiel (Fourier)Parra **and** Tobar 2017[4]
- Famille des *kernels stationnaires*, $k(t, t')$ ne dépend que de $t - t' \leftrightarrow$ "plus les inputs sont éloignées, moins les sorties associées sont corrélées"



[4] ex: $\sum_{q=0}^{Q} \sigma_q^2 e^{-2\pi^2 \Sigma_q (t-t')^2} \cos(2\pi \mu_q (t - t'))$

Le kernel

- Fléxibilité des kernels: $k_1 + k_2$, $k_1 \times k_2$, $g(t) \times k$ (g est déterministe)

Le kernel

- Fléxibilité des kernels: $k_1 + k_2$, $k_1 \times k_2$, $g(t) \times k$ (g est déterministe)
  - $k_1 + k_2$: "Ou" logique
  - $k_1 \times k_2$: "Et" logique
- Depuis quelques années: construction de kernels par des deep-nets Wilson **andothers** 2016, pratique mais peu interprétable

Le kernel

- Fléxibilité des kernels: $k_1 + k_2$, $k_1 \times k_2$, $g(t) \times k$ (g est déterministe)
  - $k_1 + k_2$: "Ou" logique
  - $k_1 \times k_2$: "Et" logique
- Depuis quelques années: construction de kernels par des deep-nets Wilson **andothers** 2016, pratique mais peu interprétable
- Plus récemment: construction de kernel à partir d'équations différentielles

Prédire avec un *GP*

Règle de Bayes

- Règle de Bayes: $p(y_2|y_1) = \frac{p(y_1,y_2)}{p(y_1)} = \frac{p(y_1|y_2)p(y_2)}{p(y_1)}$

Règle de Bayes

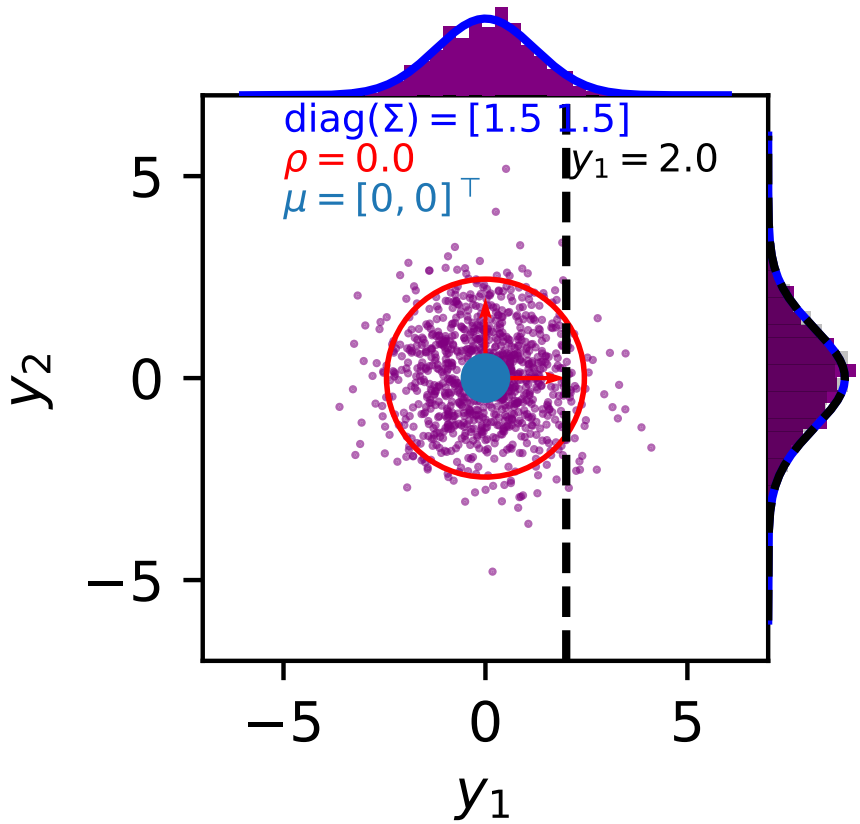
- Règle de Bayes: $p(y_2|y_1) = \frac{p(y_1,y_2)}{p(y_1)} = \frac{p(y_1|y_2)p(y_2)}{p(y_1)}$
- $\mathbf{y} = [y_1, y_2]^\top \sim \mathcal{N}(\begin{pmatrix}\mu_1 \\ \mu_2\end{pmatrix}, \begin{pmatrix}\Sigma_1 & \rho \\ \rho & \Sigma_2\end{pmatrix})$

Règle de Bayes

- Règle de Bayes: $p(y_2|y_1) = \frac{p(y_1,y_2)}{p(y_1)} = \frac{p(y_1|y_2)p(y_2)}{p(y_1)}$
- $\mathbf{y} = [y_1, y_2]^\top \sim \mathcal{N}(\begin{pmatrix}\mu_1 \\ \mu_2\end{pmatrix}, \begin{pmatrix}\Sigma_1 & \rho \\ \rho & \Sigma_2\end{pmatrix})$
- $p(y_2|y_1 = u) = \mathcal{N}(\mu_2 + \rho\Sigma_1^{-1}(u - \mu_1), \Sigma_2 - \rho^2\Sigma_1^{-1})$

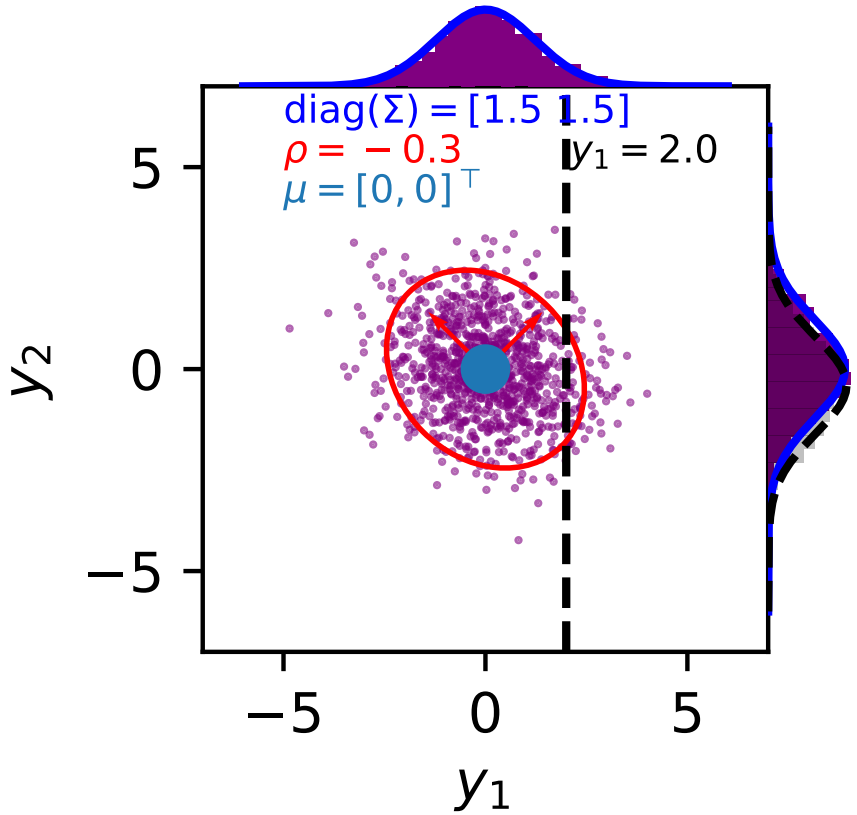
Règle de Bayes

- Règle de Bayes: $p(y_2|y_1) = \frac{p(y_1,y_2)}{p(y_1)} = \frac{p(y_1|y_2)p(y_2)}{p(y_1)}$
- $\mathbf{y} = [y_1, y_2]^\top \sim \mathcal{N}(\begin{pmatrix}\mu_1 \\ \mu_2\end{pmatrix}, \begin{pmatrix}\Sigma_1 & \rho \\ \rho & \Sigma_2\end{pmatrix})$
- $p(y_2|y_1 = u) = \mathcal{N}(\mu_2 + \rho\Sigma_1^{-1}(u - \mu_1), \Sigma_2 - \rho^2\Sigma_1^{-1})$



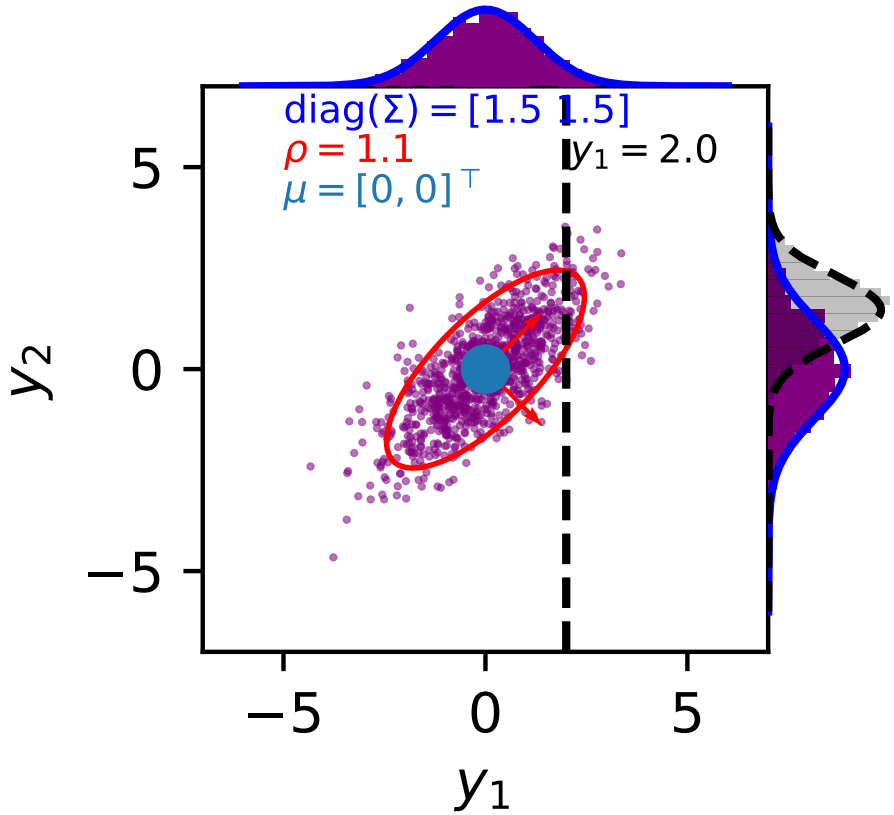
Règle de Bayes

- Règle de Bayes: $p(y_2|y_1) = \frac{p(y_1,y_2)}{p(y_1)} = \frac{p(y_1|y_2)p(y_2)}{p(y_1)}$
- $\mathbf{y} = [y_1, y_2]^\top \sim \mathcal{N}(\begin{pmatrix}\mu_1 \\ \mu_2\end{pmatrix}, \begin{pmatrix}\Sigma_1 & \rho \\ \rho & \Sigma_2\end{pmatrix})$
- $p(y_2|y_1 = u) = \mathcal{N}(\mu_2 + \rho\Sigma_1^{-1}(u - \mu_1), \Sigma_2 - \rho^2\Sigma_1^{-1})$



Règle de Bayes

- Règle de Bayes: $p(y_2|y_1) = \frac{p(y_1,y_2)}{p(y_1)} = \frac{p(y_1|y_2)p(y_2)}{p(y_1)}$
- $\mathbf{y} = [y_1, y_2]^\top \sim \mathcal{N}(\begin{pmatrix}\mu_1 \\ \mu_2\end{pmatrix}, \begin{pmatrix}\Sigma_1 & \rho \\ \rho & \Sigma_2\end{pmatrix})$
- $p(y_2|y_1 = u) = \mathcal{N}(\mu_2 + \rho\Sigma_1^{-1}(u - \mu_1), \Sigma_2 - \rho^2\Sigma_1^{-1})$



Règle de Bayes

- Règle de Bayes: $p(y_2|y_1) = \frac{p(y_1,y_2)}{p(y_1)} = \frac{p(y_1|y_2)p(y_2)}{p(y_1)}$
- $\mathbf{y} = [y_1, y_2]^\top \sim \mathcal{N}(\begin{pmatrix}\mu_1 \\ \mu_2\end{pmatrix}, \begin{pmatrix}\Sigma_1 & \rho \\ \rho & \Sigma_2\end{pmatrix})$

$$p(y_2|y_1 = u) = \mathcal{N}(\mu_2 + \rho\Sigma_1^{-1}(u - \mu_1), \underbrace{\Sigma_2 - \rho^2\Sigma_1^{-1}}_{\geq 0})$$

Conditionner $y_2|y_1 \leftrightarrow$ réduire l'incertitude (variance) de y_2
Conditionnement entre variables Gaussiennes: reste Gaussien (conjugué)

Prédire avec un *GP*

prédiction = data $\times$ hypothèse

- Data entraînement: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$, test: y_0, x_0

prédiction = data $\times$ hypothèse

- Data entraînement: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$, test: y_0, x_0
- Modèle de prédiction (régression): $y_0 = f_\theta(x_0)$

prédiction = data $\times$ hypothèse

- Data entraînement: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$, test: y_0, x_0
- Modèle de prédiction (régression): $y_0 = f_\theta(x_0)$
- Hypothèses: $f_\theta \sim GP(0, k_\theta)$ et $y_i = f_\theta(x_i)$:

$$\underbrace{[y, f_\theta(x_0)]^\top}_{(n+1)\times 1} \sim \mathcal{N}(\mathbf{0}, \begin{pmatrix} \mathbf{K}_y & \mathbf{k}_0 \\ \mathbf{k}_0^\top & k_\theta(x_0, x_0) \end{pmatrix})$$

prédiction = data × hypothèse

- Data entraînement: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$, test: y_0, x_0
- Modèle de prédiction (régression): $y_0 = f_\theta(x_0)$
- Hypothèses: $f_\theta \sim GP(0, k_\theta)$ et $y_i = f_\theta(x_i)$:

$$\underbrace{[y, f_\theta(x_0)]^\top}_{(n+1)\times 1} \sim \mathcal{N}(\mathbf{0}, \begin{pmatrix} \mathbf{K}_y & \mathbf{k}_0 \\ \mathbf{k}_0^\top & k_\theta(x_0, x_0) \end{pmatrix})$$

$$\mathbf{K}_y = \begin{pmatrix} k_\theta(x_1, x_1) & \dots & k_\theta(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k_\theta(x_n, x_1) & \dots & k_\theta(x_n, x_n) \end{pmatrix}$$

prédiction = data $\times$ hypothèse

- Data entraînement: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$, test: y_0, x_0
- Modèle de prédiction (régression): $y_0 = f_\theta(x_0)$
- Hypothèses: $f_\theta \sim GP(0, k_\theta)$ et $y_i = f_\theta(x_i)$:

$$\underbrace{[y, f_\theta(x_0)]^\top}_{(n+1)\times 1} \sim \mathcal{N}(\mathbf{0}, \begin{pmatrix} \mathbf{K}_y & \mathbf{k}_0 \\ \mathbf{k}_0^\top & k_\theta(x_0, x_0) \end{pmatrix})$$

$$\mathbf{k}_0 = [k_\theta(x_0, x_1), \ldots, k_\theta(x_0, x_n)]^\top$$

prédiction = data × hypothèse

- Data entraînement: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$, test: y_0, x_0
- Modèle de prédiction (régression): $y_0 = f_\theta(x_0)$
- Hypothèses: $f_\theta \sim GP(0, k_\theta)$ et $y_i = f_\theta(x_i)$:

$$\underbrace{[y, f_\theta(x_0)]^\top}_{(n+1)\times 1} \sim \mathcal{N}(\mathbf{0}, \begin{pmatrix} \mathbf{K}_y & \mathbf{k}_0 \\ \mathbf{k}_0^\top & k_\theta(x_0, x_0) \end{pmatrix})$$

- But: aprendre $f_\theta \rightarrow$ prédire $f_\theta(x_0) = y_0$
- Intuitivement: générer des *prédictions* avec $f_\theta \sim GP(0, k_\theta)$ ET garder ce qui fit à $[y, \mathbf{x}]$ et à x_0:

prédiction = data × hypothèse

- Data entraînement: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$, test: y_0, x_0
- Modèle de prédiction (régression): $y_0 = f_\theta(x_0)$
- Hypothèses: $f_\theta \sim GP(0, k_\theta)$ et $y_i = f_\theta(x_i)$:

$$\underbrace{[y, f_\theta(x_0)]^\top}_{(n+1)\times 1} \sim \mathcal{N}(\mathbf{0}, \begin{pmatrix} \mathbf{K}_y & \mathbf{k}_0 \\ \mathbf{k}_0^\top & k_\theta(x_0, x_0) \end{pmatrix})$$

- But: aprendre $f_\theta \rightarrow$ prédire $f_\theta(x_0) = y_0$
- Intuitivement: générer des *prédictions* avec $f_\theta \sim GP(0, k_\theta)$ ET garder ce qui fit à $[y, \mathbf{x}]$ et à x_0:

$$\underbrace{p(\textit{hypothèse}|\textit{data})}_{\textit{posterior}} = \frac{\overbrace{p(\textit{data}|\textit{hypothèse})}^{\textit{likelihood}}\overbrace{p(\textit{hypothèse})}^{\textit{prior}}}{\underbrace{p(\textit{data})}_{\textit{evidence}}}$$

prédiction = data × hypothèse

- Data entraînement: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$, test: y_0, x_0
- Modèle de prédiction (régression): $y_0 = f_\theta(x_0)$
- Hypothèses: $f_\theta \sim GP(0, k_\theta)$ et $y_i = f_\theta(x_i)$:

$$\underbrace{[y, f_\theta(x_0)]^\top}_{(n+1)\times 1} \sim \mathcal{N}(\mathbf{0}, \begin{pmatrix} \mathbf{K}_y & \mathbf{k}_0 \\ \mathbf{k}_0^\top & k_\theta(x_0, x_0) \end{pmatrix})$$

- But: aprendre $f_\theta \rightarrow$ prédire $f_\theta(x_0) = y_0$
- Intuitivement: générer des *prédictions* avec $f_\theta \sim GP(0, k_\theta)$ ET garder ce qui fit à $[y, \mathbf{x}]$ et à x_0:
- Revient à générer selon:

$$\overbrace{p(f_\theta(\mathbf{x}), f_\theta(x_0)|y, \mathbf{x}, x_0)}^{posterior}$$

prédiction = data × hypothèse

- Data entraînement: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$, test: y_0, x_0
- Modèle de prédiction (régression): $y_0 = f_\theta(x_0)$
- Hypothèses: $f_\theta \sim GP(0, k_\theta)$ et $y_i = f_\theta(x_i)$:

$$\underbrace{[y, f_\theta(x_0)]^\top}_{(n+1)\times 1} \sim \mathcal{N}(\mathbf{0}, \begin{pmatrix} \mathbf{K}_y & \mathbf{k}_0 \\ \mathbf{k}_0^\top & k_\theta(x_0, x_0) \end{pmatrix})$$

- But: aprendre $f_\theta \rightarrow$ prédire $f_\theta(x_0) = y_0$
- Intuitivement: générer des *prédictions* avec $f_\theta \sim GP(0, k_\theta)$ ET garder ce qui fit à $[y, \mathbf{x}]$ et à x_0:
- Revient à générer selon:

$$\overbrace{p(f_\theta(\mathbf{x}), f_\theta(x_0)|y, \mathbf{x}, x_0)}^{posterior} = \frac{\overbrace{p(y|f_\theta(\mathbf{x}))}^{likelihood}\overbrace{p(f_\theta(\mathbf{x}), f_\theta(x_0))}^{prior}}{p(y|\mathbf{x})} = GP(m_*, k_{*,\theta})$$

prédiction = data $\times$ hypothèse

Prédiction (aka inférence):

- $m_*, k_{*,\theta}$?
- $p(f_\theta | y, \mathbf{x}) = GP(m_*, k_{*,\theta})$

prédiction = data $\times$ hypothèse

Prédiction (aka inférence):

- $m_*, k_{*,\theta}$?
- $p(f_\theta | y, \mathbf{x}) = GP(m_*, k_{*,\theta})$
- $m_*(x_0) = 0 + \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} y$

prédiction = data $\times$ hypothèse

Prédiction (aka inférence):

- $m_*, k_{*,\theta}$?
- $p(f_\theta | y, \mathbf{x}) = GP(m_*, k_{*,\theta})$
- $m_*(x_0) = 0 + \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} y$

prédiction = data $\times$ hypothèse

Prédiction (aka inférence):

- $m_*, k_{*,\theta}$?
- $p(f_\theta|y, \mathbf{x}) = GP(m_*, k_{*,\theta})$
- $m_*(x_0) = 0 + \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} y$
- $k_{*,\theta}(x_0, x_0') = k_{\theta(x_0, x_0')-}$

prédiction = data $\times$ hypothèse

Prédiction (aka inférence):

- $m_*, k_{*,\theta}$?
- $p(f_\theta | y, \mathbf{x}) = GP(m_*, k_{*,\theta})$
- $m_*(x_0) = 0 + \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} y$
- $k_{*,\theta}(x_0, x_0') = k_{\theta(x_0,x_0')} - \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} \mathbf{k}_0$

prédiction = data × hypothèse

Prédiction (aka inférence):

- $m_*, k_{*,\theta}$?
- $p(f_\theta|y, \mathbf{x}) = GP(m_*, k_{*,\theta})$
- $m_*(x_0) = 0 + \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} y$
- $k_{*,\theta}(x_0, x_0') = k_{\theta(x_0,x_0')} - \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} \mathbf{k}_0$
- La distribution de probabilités de la prédiction y_0 (connaissant les data d'entraînement) est caractérisée !
- Pour n'importe quel x_0: accès à la prédiction moyenne, incertitude (intervalle de confiance), quantiles, etc., de y_0

prédiction = data × hypothèse

Prédiction (aka inférence):

- $m_*, k_{*,\theta}$?
- $p(f_\theta|y, \mathbf{x}) = GP(m_*, k_{*,\theta})$
- $m_*(x_0) = 0 + \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} y$
- $k_{*,\theta}(x_0, x_0') = k_{\theta(x_0,x_0')} - \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} \mathbf{k}_0$
- La distribution de probabilités de la prédiction y_0 (connaissant les data d'entraînement) est caractérisée !
- Pour n'importe quel x_0: accès à la prédiction moyenne, incertitude (intervalle de confiance), quantiles, etc., de y_0
- Modèle de prédiction = { data d'entraînement, hyperparamètres }

prédiction = data × hypothèse

Prédiction (aka inférence):

- $m_*, k_{*,\theta}$?
- $p(f_\theta | y, \mathbf{x}) = GP(m_*, k_{*,\theta})$
- $m_*(x_0) = 0 + \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} y$
- $k_{*,\theta}(x_0, x_0') = k_{\theta(x_0,x_0')} - \mathbf{k}_0^\top \mathbf{K}_y{}^{-1} \mathbf{k}_0$
- La distribution de probabilités de la prédiction y_0 (connaissant les data d'entraînement) est caractérisée !
- Pour n'importe quel x_0: accès à la prédiction moyenne, incertitude (intervalle de confiance), quantiles, etc., de y_0
- Modèle de prédiction = { data d'entraînement, hyperparamètres }
- La prédiction dépend aussi de θ (hyperparamètre) !

prédiction = data $\times$ hypothèse

Entraînement:

- Meilleur θ ?

prédiction = data $\times$ hypothèse

Entraînement:

- Meilleur θ ?
- Maximum de *likelihood*$(\theta) := p(y|f_\theta(\mathbf{x}))$?

prédiction = data $\times$ hypothèse

Entraînement:

- Meilleur θ ?
- Maximum de *likelihood*$(\theta) := p(y|f_\theta(\mathbf{x}))$?
- NON !! Restreint l'hypothèse uniquement à $[y, \mathbf{x}]$ (overfitting)

prédiction = data $\times$ hypothèse

Entraînement:

- Meilleur θ ?
- Maximum de *likelihood*$(\theta) := p(y|f_\theta(\mathbf{x}))$?
- NON !! Restreint l'hypothèse uniquement à $[y, \mathbf{x}]$ (overfitting)
- Mieux: hypothèse sur les data + voisinage
- Maximum de *likelihood marginale*:

prédiction = data × hypothèse

Entraînement:

- Meilleur θ ?
- Maximum de *likelihood*$(\theta) := p(y|f_\theta(\mathbf{x}))$?
- NON !! Restreint l'hypothèse uniquement à $[y, \mathbf{x}]$ (overfitting)
- Mieux: hypothèse sur les data + voisinage
- Maximum de *likelihood marginale*:

$$= p(y|f_\theta(\mathbf{x}))p(f_\theta(\mathbf{x}))$$

- $\leftrightarrow \theta$ telle que les données d'entraînement y soient le plus probables au regard de toutes les des hypothèses possibles et des inputs $\mathbf{x}$

prédiction = data $\times$ hypothèse

Entraînement:

- Meilleur θ ?
- Maximum de *likelihood*$(\theta) := p(y|f_\theta(\mathbf{x}))$?
- NON !! Restreint l'hypothèse uniquement à $[y, \mathbf{x}]$ (overfitting)
- Mieux: hypothèse sur les data + voisinage
- Maximum de *likelihood marginale*:

$$p(y|\mathbf{x}, \theta) = \int_{\mathbb{R}^n} p(y|\mathbf{x}, f_\theta(\mathbf{x}))p(f_\theta(\mathbf{x}))\mathrm{d}f_\theta$$

- $\leftrightarrow \theta$ telle que les données d'entraînement y soient le plus probables au regard de toutes les des hypothèses possibles et des inputs $\mathbf{x}$

prédiction = data × hypothèse

Entraînement:

- Meilleur θ ?
- Maximum de *likelihood*$(\theta) := p(y|f_\theta(\mathbf{x}))$?
- NON !! Restreint l'hypothèse uniquement à $[y, \mathbf{x}]$ (overfitting)
- Mieux: hypothèse sur les data + voisinage
- Maximum de *likelihood marginale*:

$$p(y|\mathbf{x}, \theta) = \int_{\mathbb{R}^n} p(y|\mathbf{x}, f_\theta(\mathbf{x}))p(f_\theta(\mathbf{x}))\mathrm{d}f_\theta$$
$$= \int \mathcal{N}(f_\theta(\mathbf{x}), I_d)\mathcal{N}(\mathbf{0}, \mathbf{K}_y)\mathrm{d}f_\theta$$

- $\leftrightarrow \theta$ telle que les données d'entraînement y soient le plus probables au regard de toutes les des hypothèses possibles et des inputs $\mathbf{x}$

prédiction = data × hypothèse

$$\begin{aligned}\log \textit{likelihoodmarg}(\theta) &= \log p(y|\mathbf{x},\theta) (= \log \prod_i p(y_i|x_i,\theta)) \\ &= \log \mathcal{N}(\mathbf{0}, \mathbf{K}_y + I_d) \\ &= \underbrace{-y^\top \mathbf{K}_y{}^{-1} y}_{\textit{fidélité}} \underbrace{- \frac{1}{2} \log |\mathbf{K}_y|}_{\textit{pénalité}} - \frac{n}{2} \log 2\pi\end{aligned}$$

prédiction = data × hypothèse

- $$\begin{aligned}\log \textit{likelihoodmarg}(\theta) &= \log p(y|\mathbf{x},\theta) (= \log \prod_i p(y_i|x_i,\theta)) \\ &= \log \mathcal{N}(\mathbf{0}, \mathbf{K}_y + I_d) \\ &= -\underbrace{y^\top \mathbf{K}_y{}^{-1} y}_{\textit{fidélité}} - \underbrace{\frac{1}{2}\log|\mathbf{K}_y|}_{\textit{pénalité}} - \frac{n}{2}\log 2\pi\end{aligned}$$

- Rappel: $\mathbf{K}_y$ matrice $n \times n$ des $k_\theta(x_i, x_j) = \mathbb{C}(f_\theta(x_i), f_\theta(x_j)) \overset{ex}{=} e^{-\frac{(x_i - x_j)^2}{2\ell^2}}$

prédiction = data × hypothèse

$$\begin{aligned}\log \textit{likelihoodmarg}(\theta) &= \log p(y|\mathbf{x},\theta) (= \log \prod_i p(y_i|x_i,\theta)) \\ &= \log \mathcal{N}(\mathbf{0}, \mathbf{K}_y + I_d) \\ &= -\underbrace{y^\top \mathbf{K}_y{}^{-1} y}_{\textit{fidélité}} - \underbrace{\frac{1}{2}\log|\mathbf{K}_y|}_{\textit{pénalité}} - \frac{n}{2}\log 2\pi\end{aligned}$$

- Rappel: $\mathbf{K}_y$ matrice $n \times n$ des $k_\theta(x_i, x_j) = \mathbb{C}(f_\theta(x_i), f_\theta(x_j)) \overset{ex}{=} e^{-\frac{(x_i - x_j)^2}{2\ell^2}}$
- Apprentissage de θ par descente de gradient de $-\log \textit{likelihoodmarg}$ (large sous-domaine de la communauté GP)

Prédire avec un *GP*

Méthodo générale:

1. Collecter des données: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$

Prédire avec un *GP*

Méthodo générale:

1. Collecter des données: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$
2. Formuler des hypothèses: choisir/combiner un/des kernels k_θ

Prédire avec un *GP*

Méthodo générale:

1. Collecter des données: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$
2. Formuler des hypothèses: choisir/combiner un/des kernels k_θ
3. Entraîner le modèle: estimer les hyperparamètres θ

Prédire avec un *GP*

Méthodo générale:

1. Collecter des données: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$
2. Formuler des hypothèses: choisir/combiner un/des kernels k_θ
3. Entraîner le modèle: estimer les hyperparamètres θ
4. Prédire ("*inférence*"): évaluer m_* et $k_{*,\theta}$, générer des données, etc.

Prédire avec un *GP*

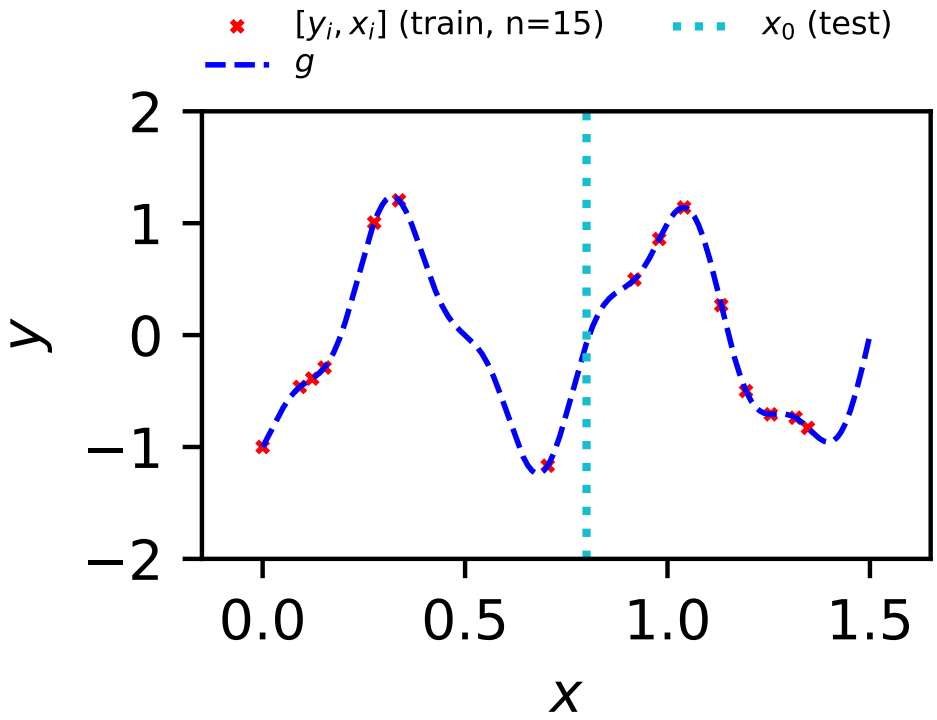
Méthodo générale:

1. Collecter des données: $\{y_i, x_i\}_{1 \leq i \leq n} = [y, \mathbf{x}]$
2. Formuler des hypothèses: choisir/combiner un/des kernels k_θ
3. Entraîner le modèle: estimer les hyperparamètres θ
4. Prédire ("*inférence*"): évaluer m_* et $k_{*,\theta}$, générer des données, etc.
5. Critique/évaluation du modèle

Prédire avec un *GP*

Exemples

- train $= \{y_i, x_i\}_{i \leq n}$ où $y_i = g(x_i) = -\cos(3\pi x_i) + \frac{1}{4}\sin(8\pi x_i)$
- g inconnue



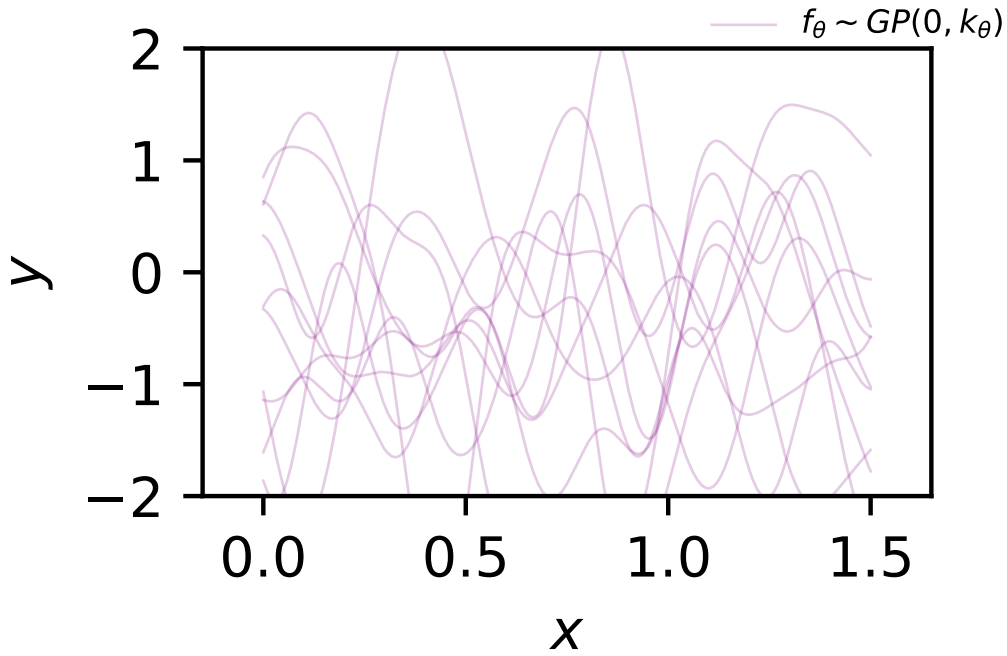
- $f_\theta(x_0 = 0.8) = \tilde{y}_0$? (*interpolation*)

Exemples

- Hypothèses: $y_i = f(x_i)$ et f est lisse $\rightarrow$ kernel exponentiel quadratique
- $k_\theta(x, x') = \sigma_k^2 e^{-\frac{(x-x')^2}{2\ell^2}}, \theta = [\sigma_k, \ell]$

Exemples

- Hypothèses: $y_i = f(x_i)$ et f est lisse $\rightarrow$ kernel exponentiel quadratique
- $k_\theta(x, x') = \sigma_k^2 e^{-\frac{(x-x')^2}{2\ell^2}}, \theta = [\sigma_k, \ell]$
- $[\sigma_k, \ell]$ inconnus (ici fixés arbitrairement)



Exemples

Entraînement

- θ ? minimisation de $-$ log likelihood marginale: $\sigma_k \approx 0.92, \ell \approx 0.11$

Exemples

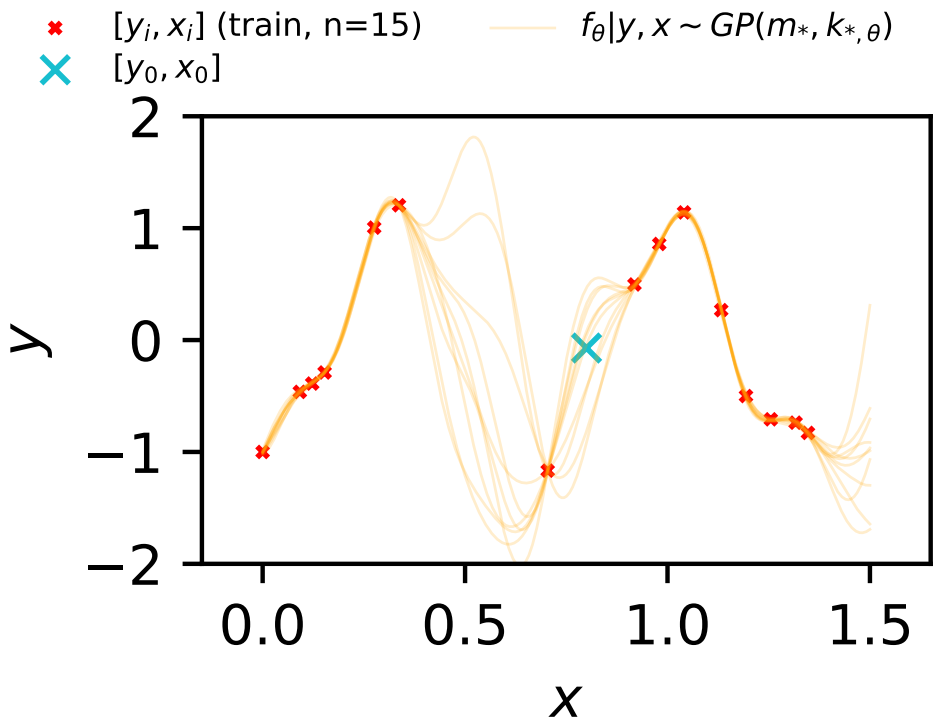
Entraînement

- θ ? minimisation de $-$ log likelihood marginale: $\sigma_k \approx 0.92, \ell \approx 0.11$
- $m_*, k_{*,\theta}$ et θ appris

Exemples

Entraînement

- θ ? minimisation de $-\log$ likelihood marginale: $\sigma_k \approx 0.92, \ell \approx 0.11$
- $m_*, k_{*,\theta}$ et θ appris
- Inférence: générer selon la distribution posterior $GP(m_*, k_{*,\theta})$



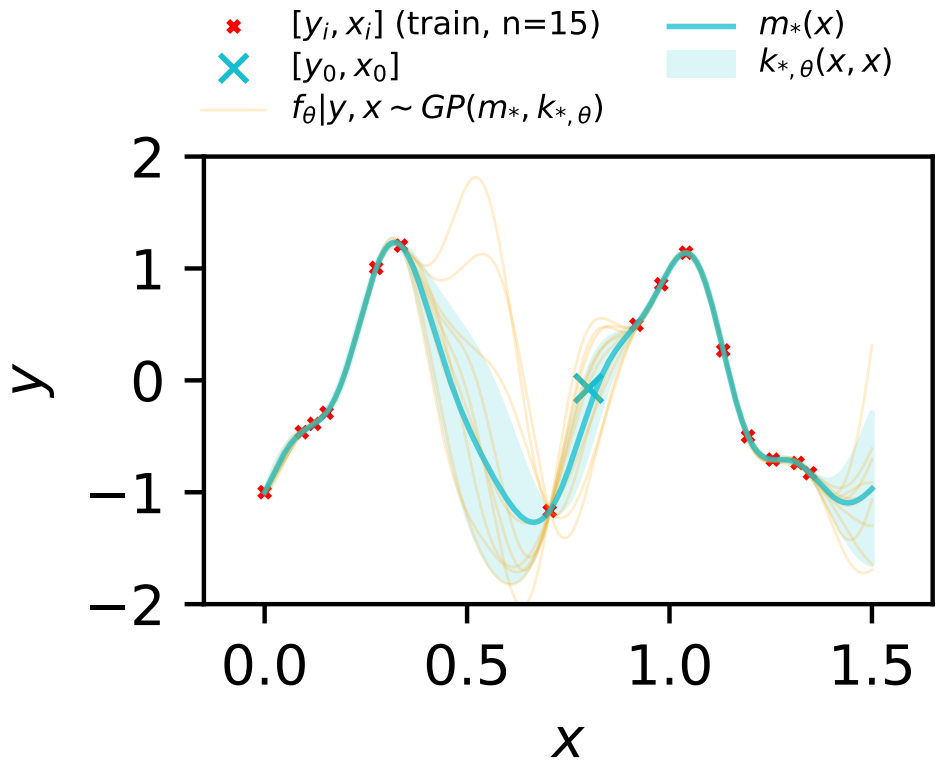
Exemples

Inférence

- $\tilde{y}_0$: moyenne de $p(f_\theta|y, \mathbf{x}, x_0) = GP(m_*, k_{*,\theta})$ en $x_0 = 0.8$
- $m_*(x_0) \pm \frac{1}{2}\sqrt{k_{*,\theta}(x_0, x_0)}$: $= -0.22 \pm 0.16$ (95%)

Exemples

- Quid d'autres valeurs de x_0 ?



- Incertitude sur "toute" la fonction

Exemples

- Hypothèses précédentes: "pas de bruit" (hypothèse forte), "lisse" (hypothèse large)

Exemples

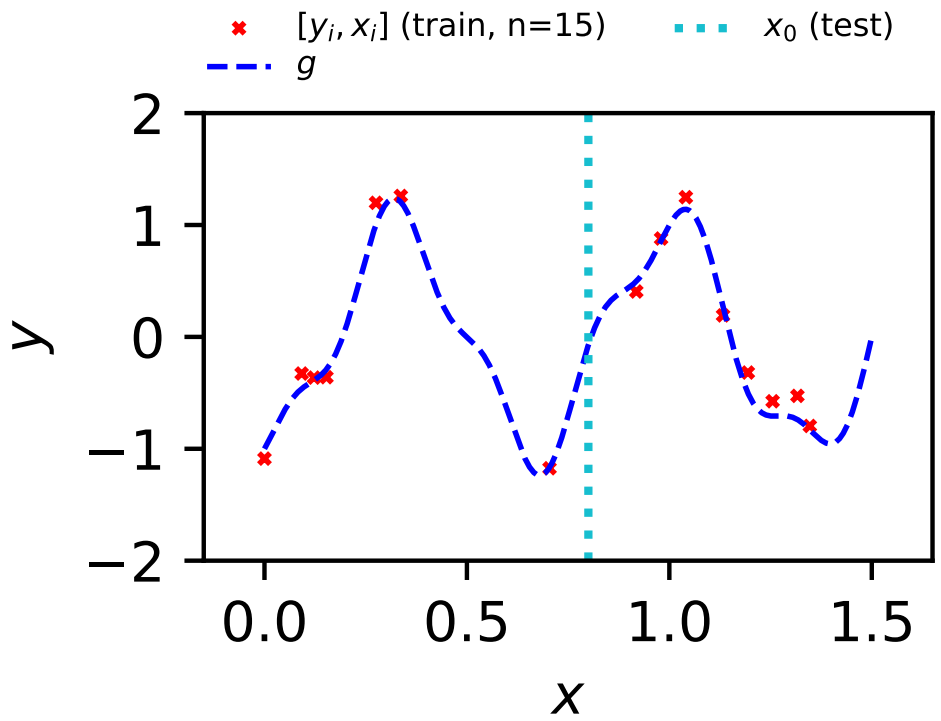
- Hypothèses précédentes: "pas de bruit" (hypothèse forte), "lisse" (hypothèse large)
- Hypothèses plus réalistes ?

Exemples

- Hypothèses précédentes: "pas de bruit" (hypothèse forte), "lisse" (hypothèse large)
- Hypothèses plus réalistes ?
- Bruit de mesure $y_i = f_\theta(x_i) + \epsilon$, lisse, quasi-périodique: $k_\theta = k_\epsilon + \sigma_{exp2} k_{exp2} \times \sigma_{per} k_{per}$
  - k_ϵ: bruit
  - $k_{exp2} \times k_{per}$: Lisse ET quasi-périodique
  - $\theta = [\sigma_\epsilon, \sigma_{exp2}, \ell, \sigma_{per}, T_0, \ell_{per}]$

Exemples

- Même données + bruit blanc ($\sigma = 0.3$)
- $f_\theta(x_0 = 0.8) = \tilde{y}_0$ (*smoothing*) ?



Exemples

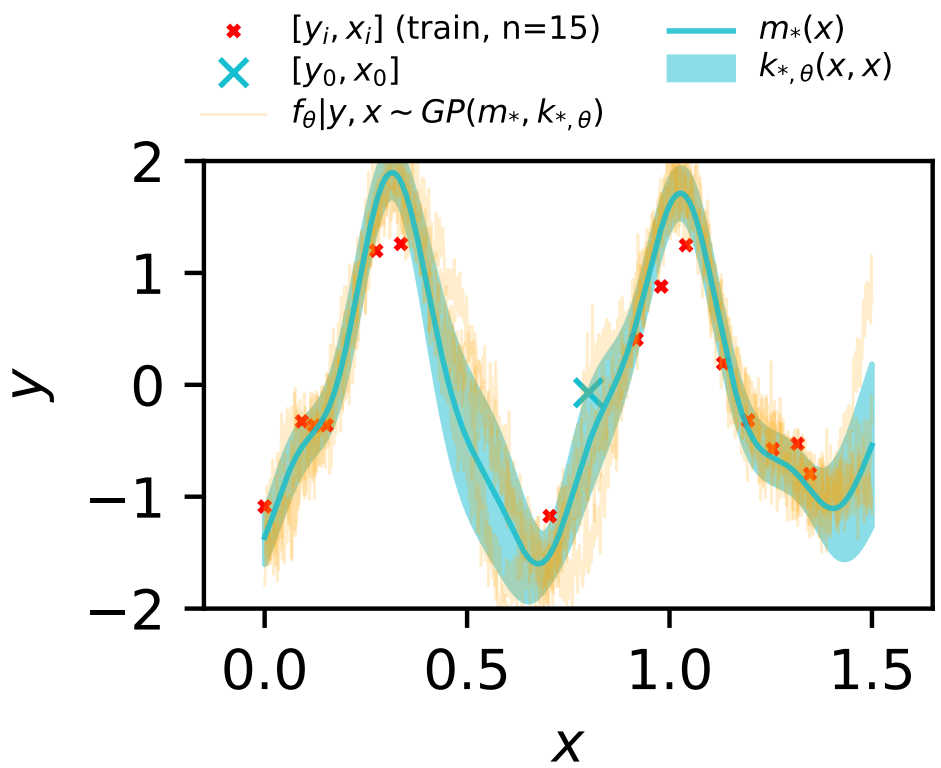
- θ ? minimisation de $-\log$ likelihood: $\sigma_\epsilon \approx 0.24$, $\sigma_{exp2} \approx 1$, $\ell \approx 1$, $\sigma_{per} \approx 1$, $T_0 = 0.7$, $\ell_{per} \approx= 1$

Exemples

- θ ? minimisation de $-\log$ likelihood: $\sigma_\epsilon \approx 0.24$, $\sigma_{exp2} \approx 1$, $\ell \approx 1$, $\sigma_{per} \approx 1$, $T_0 = 0.7$, $\ell_{per} \approx= 1$
- m_*, $k_{*,\theta}$ et θ appris

Exemples

- θ ? minimisation de $-\log$ likelihood: $\sigma_\epsilon \approx 0.24$, $\sigma_{exp2} \approx 1$, $\ell \approx 1$, $\sigma_{per} \approx 1$, $T_0 = 0.7$, $\ell_{per} \approx= 1$
- m_*, $k_{*,\theta}$ et θ appris
- Génération selon la distribution posterior $GP(m_*, k_{*,\theta})$



Exemples

- Pour $x > 1.5$? (*forecast*)

Exemples

- Pour $x > 1.5$? (*forecast*)
- Comparaison des deux hypothèses

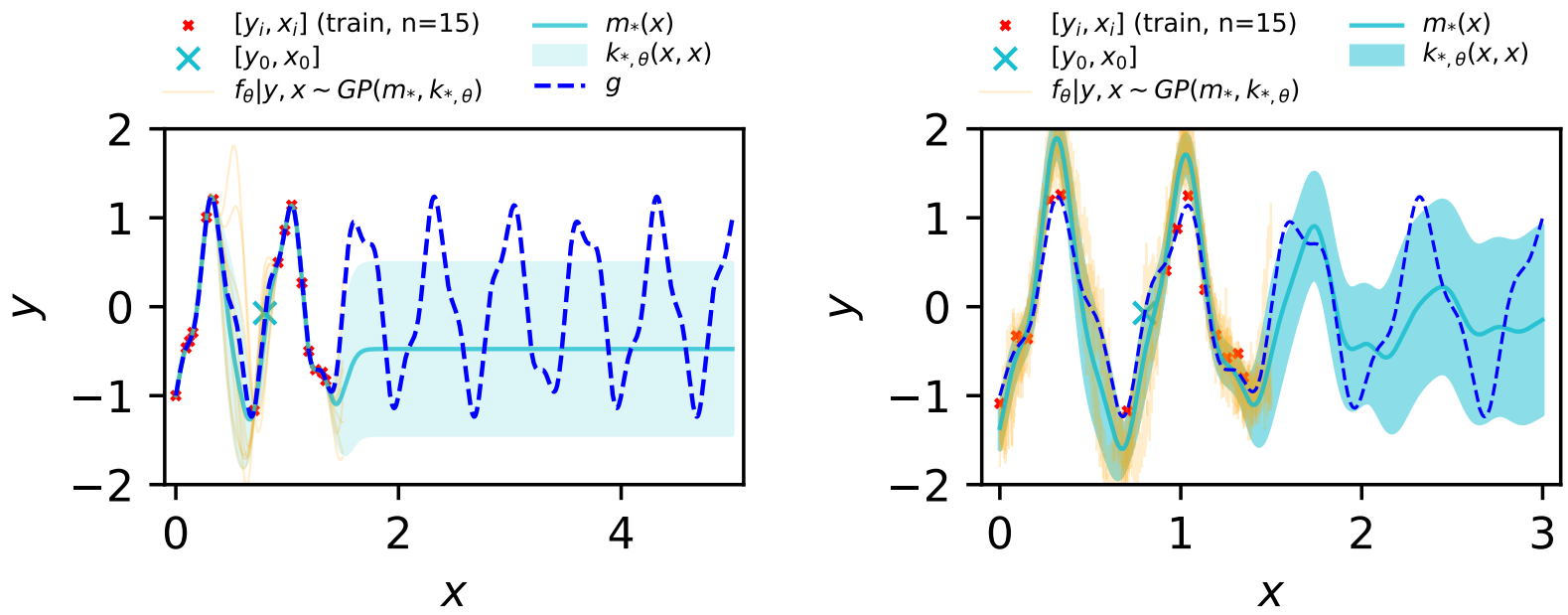
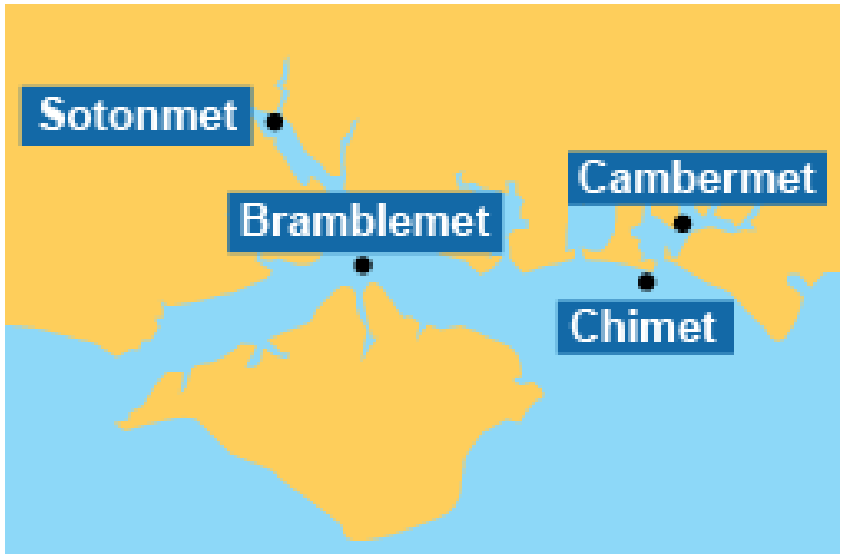


Figure: (a): sans bruit, lisse (b): bruit, lisse et périodique

Prédire avec un *GP*

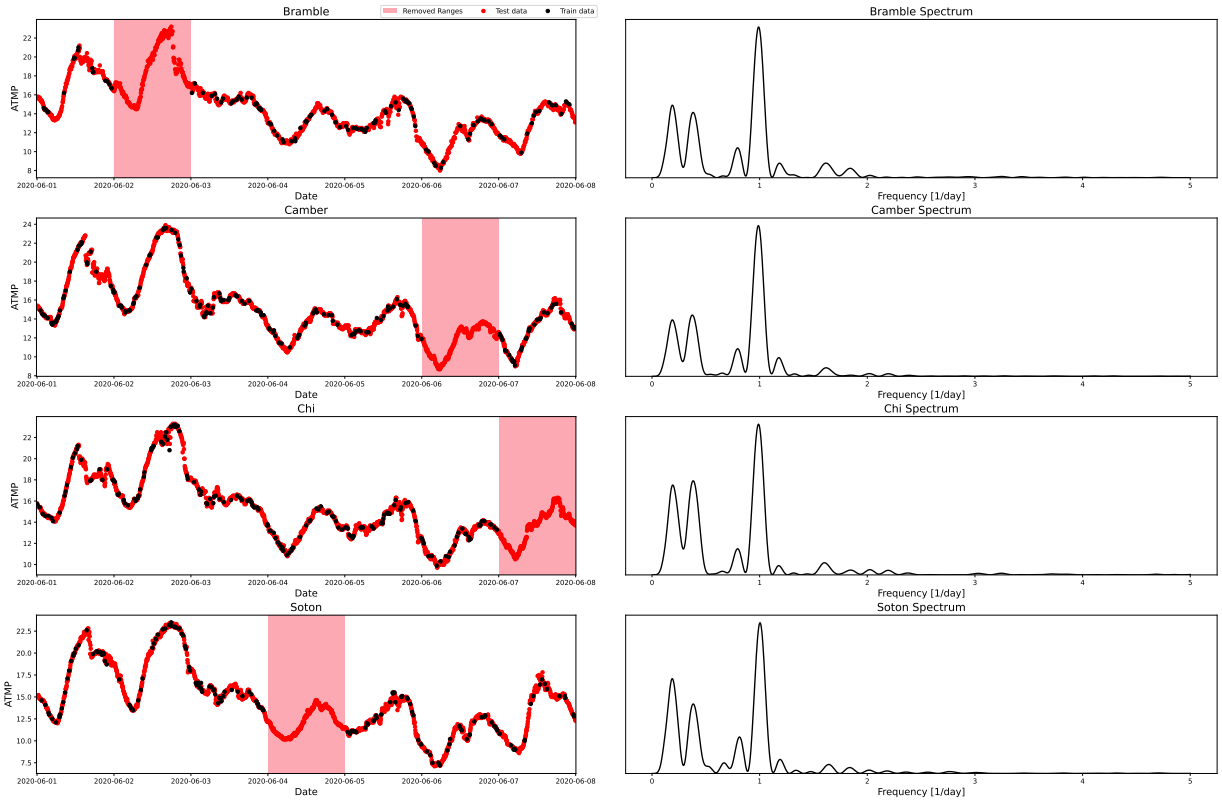
Application: prédiction de température maritime

- Séries temporelles de température sur la côté sud anglaise[5]: 4 stations sur 7 jours, $\Delta t \approx 5$min



Application: prédiction de température maritime

- Séries temporelles de température sur la côté sud anglaise[5]: 4 stations sur 7 jours, $\Delta t \approx 5\text{min}$
- But: estimation du bruit, forecasting et complétion de valeurs manquantes



[5] Accessible ici

Application: prédiction de température maritime

- Série temporelle multivariée, *GP* défini de $\mathcal{X} = \mathbb{R}^+$ (temps) vers $\mathcal{Y} = \mathbb{R}^4$ (stations)

Application: prédiction de température maritime

- Série temporelle multivariée, GP défini de $\mathcal{X} = \mathbb{R}^+$ (temps) vers $\mathcal{Y} = \mathbb{R}^4$ (stations)
- Hypothèses: les stations sont corrélées, bruitées et périodiques
- Deux niveaux de corrélation: intra-station et inter-station

$$\mathbf{K}(\mathrm{t},\, \mathrm{t}') = \begin{pmatrix} k_{11}(t,t') & k_{12}(t,t') & k_{13}(t,t') & k_{14}(t,t') \\ \vdots & k_{22}(t,t') & k_{23}(t,t') & k_{24}(t,t') \\ \vdots & \vdots & \ddots & \vdots \\ k_{41}(t,t') & \dots & \dots & k_{44}(t,t') \end{pmatrix}$$

- Chaque station doit être prédite avec par elle-même + les autres

Application: prédiction de température maritime

- Kernels "spectraux"[6] Parra **and** Tobar 2017: fit les puissances spectrales pour estimer les périodicités intra/inter et filtrer le bruit (hautes fréquences)
- Entraînement: $n = 644$, test: $n = 6845$, 84 hyperparamètres (notebook dispo: détails optim etc.)
- ...
- 1500 itérations (1min 23s)
- Comparaison avec le même kernel sans corrélation inter-station

[6] de la forme $k_{ij}(t, t') = \alpha_{ij} e^{-\frac{(|t-t'|+\theta_{ij})^2}{2\Sigma_{ij}}} \cos((|t-t'|+\theta_{ij})\mu_{ij} + \phi_{ij})$

Application: prédiction de température maritime

	MAE	MAPE	RMSE
Bramble (NaN)	2.10	10.37	2.79
Camber (NaN)	1.58	13.82	1.74
Chi (forecast)	1.76	12.53	2.01
Soton (NaN)	1.68	15.00	2.13

kernel *sans* corrélation inter-station

	MAE	MAPE	RMSE
Bramble	0.65	3.44	0.89
Camber	0.61	5.27	0.72
Chi	0.61	4.72	0.71
Soton	0.45	3.55	0.54

kernel *avec* corrélation inter-station

Table: Erreurs de prédiction

Application: prédiction de température maritime

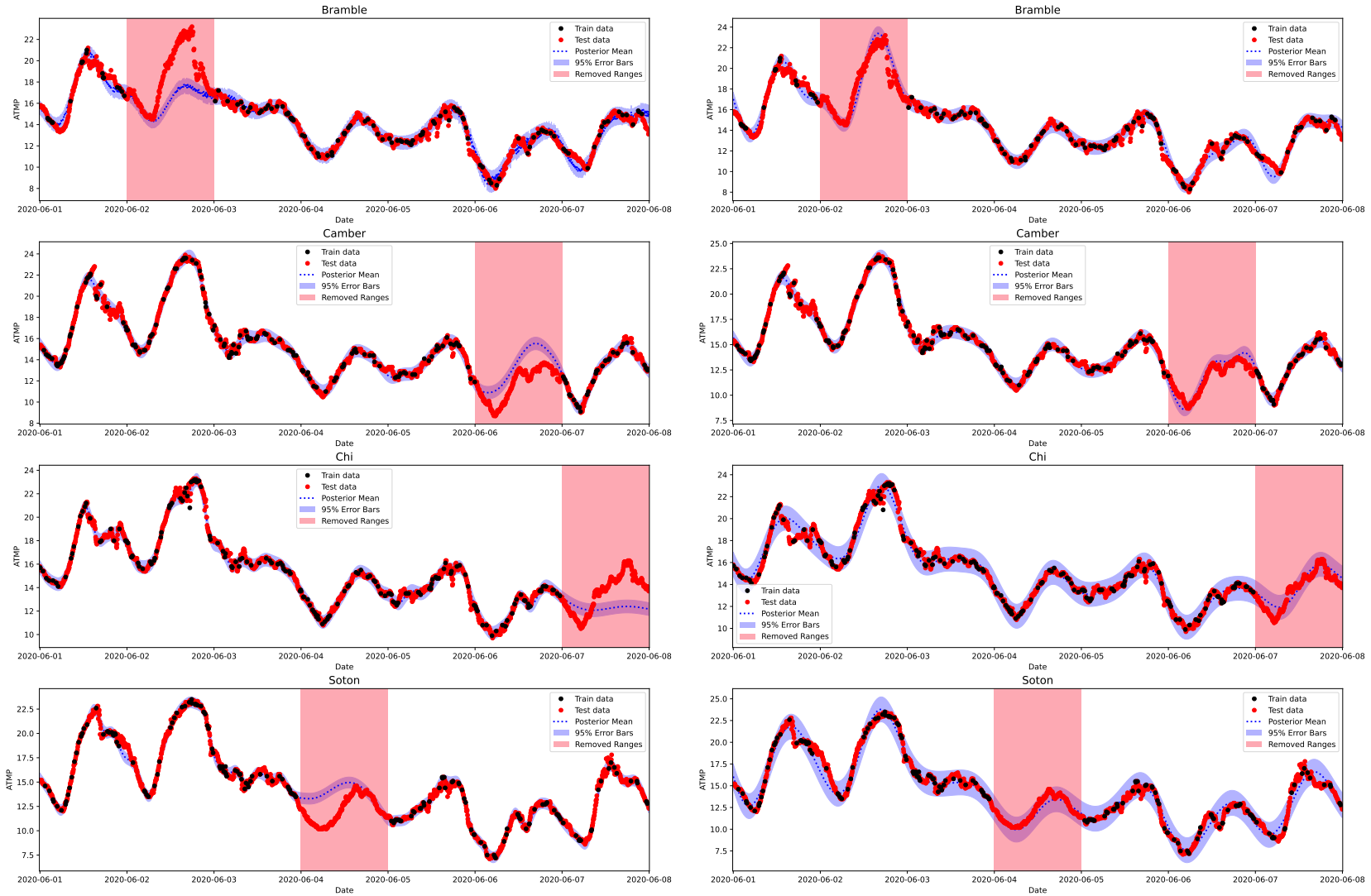


Figure: Pédictions (rouge) sans (resp avec) corrélation inter-stations à gauche (resp droite)

Application: prédiction de température maritime

- Kernel appris par chaque modèle:

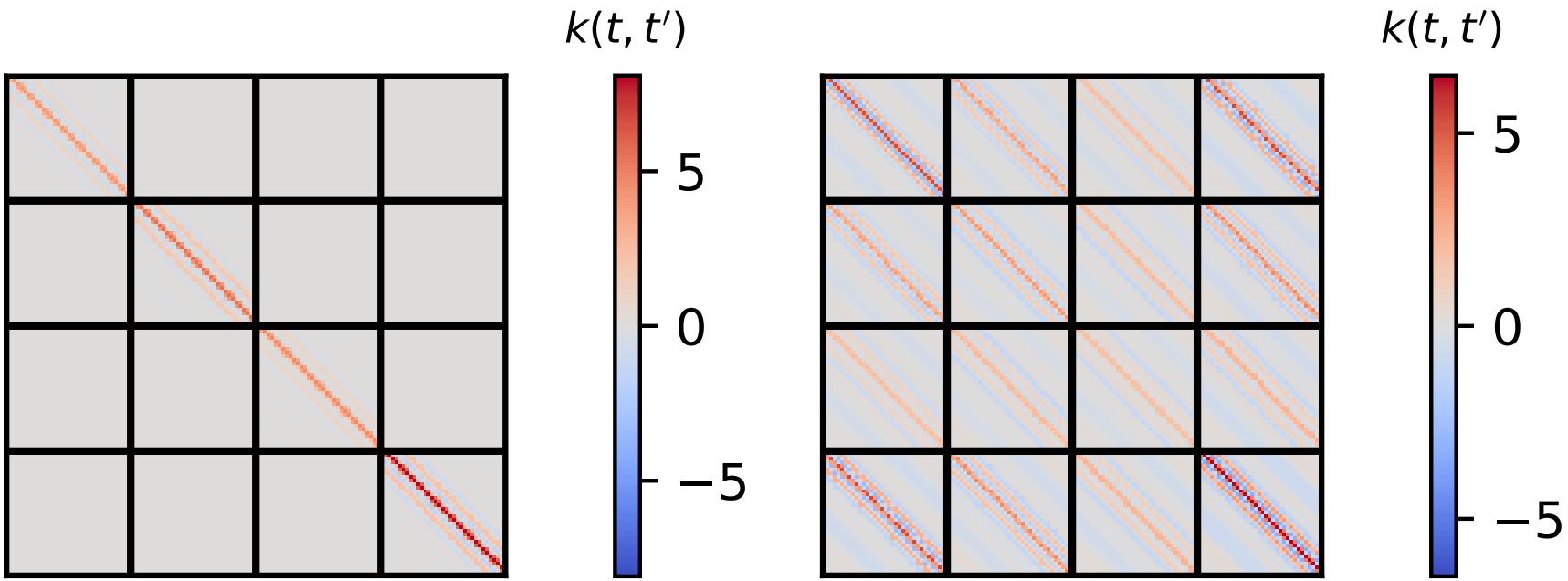


Figure: Kernel sans (resp avec) corrélation inter-stations à gauche (resp droite

Application: prédiction de température maritime

- Kernel appris par chaque modèle:

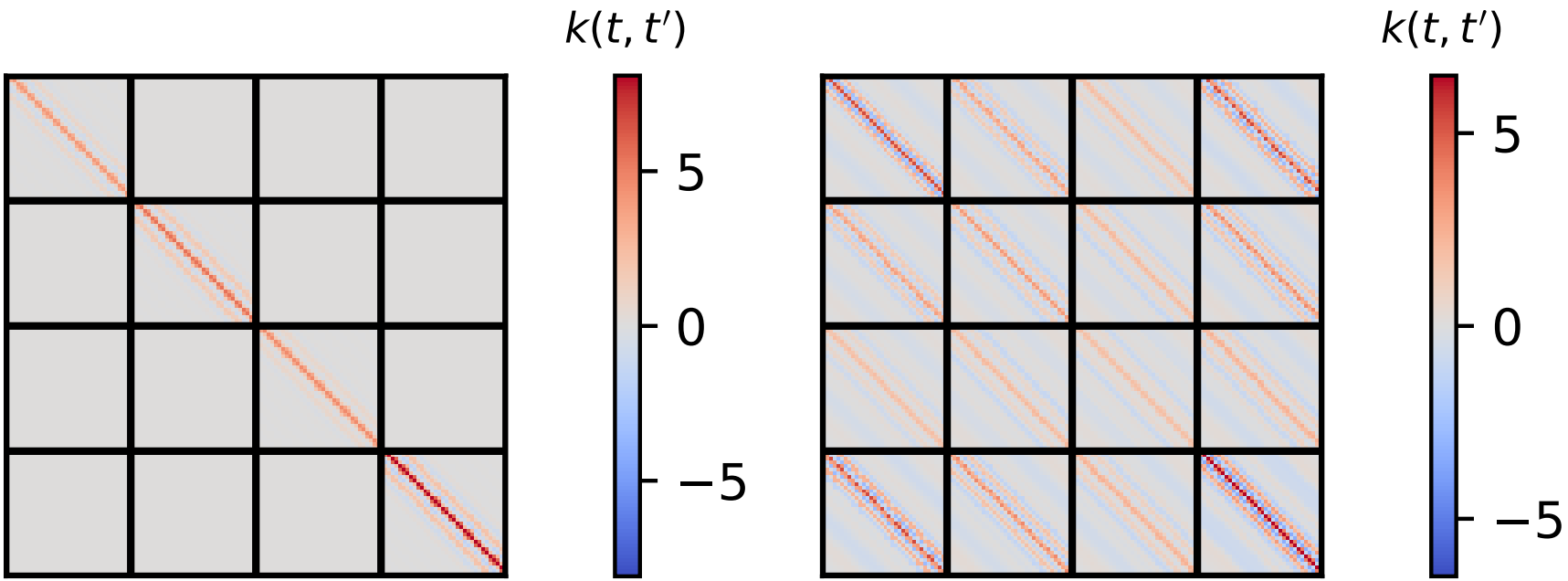


Figure: Kernel sans (resp avec) corrélation inter-stations à gauche (resp droite

- Le modèle corrèle les prédictions de chaque station à court-terme

Application: prédiction de température maritime

- Prédictions probabilistes:

Application: prédiction de température maritime

- Prédictions probabilistes:
- Quelle est la probabilité qu'il fasse entre 13° et 14° à Bramble au 08/06/2020 à minuit ?
- Prédiction du modèle: 59%

Application: prédiction de température maritime

- Prédictions probabilistes:
- Quelle est la probabilité qu'il fasse entre 13° et 14° à Bramble au 08/06/2020 à minuit ?
- Prédiction du modèle: 59%
- A l'inverse: à la même date, quel intervalle à 90% ?
- Prédiction du modèle: $[14.2; 14.9]^\circ$

Avantages / Limites des *GP* naïfs

Avantages

- *Probabiliste*: génératif, fourni une confiance (aka incertitude)

Avantages

- *Probabiliste*: génératif, fourni une confiance (aka incertitude)
- *Bayésien*: "contrôle" des hypothèses (a priori)

Avantages

- *Probabiliste*: génératif, fourni une confiance (aka incertitude)
- *Bayésien*: "contrôle" des hypothèses (a priori)
- *Non-paramétrique*: prédiction = data × hypothèse
  1. Paramètres du *GP* (moyenne, covariance) = les/des données d'entraînement, de taille infinie (non-bornée)

Avantages

- *Probabiliste*: génératif, fourni une confiance (aka incertitude)
- *Bayésien*: "contrôle" des hypothèses (a priori)
- *Non-paramétrique*: prédiction = data × hypothèse
  1. Paramètres du *GP* (moyenne, covariance) = les/des données d'entraînement, de taille infinie (non-bornée)
  2. + les hyperparamètres (ce qui règle l'hypothèse)

Avantages

- *Probabiliste*: génératif, fourni une confiance (aka incertitude)
- *Bayésien*: "contrôle" des hypothèses (a priori)
- *Non-paramétrique*: prédiction = data × hypothèse
  1. Paramètres du *GP* (moyenne, covariance) = les/des données d'entraînement, de taille infinie (non-bornée)
  2. + les hyperparamètres (ce qui règle l'hypothèse)
- Le modèle prédictif n'est pas "réduit" à un ensemble *fini* de paramètres

Avantages

- *Probabiliste*: génératif, fourni une confiance (aka incertitude)
- *Bayésien*: "contrôle" des hypothèses (a priori)
- *Non-paramétrique*: prédiction = data $\times$ hypothèse
  1. Paramètres du *GP* (moyenne, covariance) = les/des données d'entraînement, de taille infinie (non-bornée)
  2. + les hyperparamètres (ce qui règle l'hypothèse)
- Le modèle prédictif n'est pas "réduit" à un ensemble *fini* de paramètres
- Cadre naturel des données spatio/temporelles: échantillonnage irrégulier en temps ($\neq$ SARIMA) et espace, décomposition de la variabilité

Avantages

- *Probabiliste*: génératif, fourni une confiance (aka incertitude)
- *Bayésien*: "contrôle" des hypothèses (a priori)
- *Non-paramétrique*: prédiction = data × hypothèse
  1. Paramètres du *GP* (moyenne, covariance) = les/des données d'entraînement, de taille infinie (non-bornée)
  2. + les hyperparamètres (ce qui règle l'hypothèse)
- Le modèle prédictif n'est pas "réduit" à un ensemble *fini* de paramètres
- Cadre naturel des données spatio/temporelles: échantillonnage irrégulier en temps ($\neq$ SARIMA) et espace, décomposition de la variabilité
- Extensible à plus de dimensions d'entrée que le temps

Avantages / Limites des *GP* naïfs

Limites

Scalabilité:

- Bottleneck = inverser la matrice de covariance (kernel)
- La distribution prédictive $GP(m_*, k_{*,\theta})$ est lourde à calculer ($n \gg 10^4$)
  - Entraînement (log-likelihood): calcul: $\mathcal{O}(n^3)$/itération. Mémoire: $\mathcal{O}(n^2)$
  - Prédiction (inférence): calcul $\mathcal{O}(n^3)$ (une fois),

Limites

GP pour les données non-Gaussiennes ?

- Monde Gaussien: symmétrique, pas d'outliers, valeurs réelles

Limites

GP pour les données non-Gaussiennes ?

- Monde Gaussien: symmétrique, pas d'outliers, valeurs réelles

$$\overbrace{p(f_\theta|y,\mathbf{x},x_0)}^{posterior} = \frac{\overbrace{p(y|\mathbf{x},f_\theta)}^{likelihood}\ \overbrace{p(f_\theta)}^{prior=Gaussien}}{\underbrace{p(y|\mathbf{x})}_{Gaussien}}$$

Limites

GP pour les données non-Gaussiennes ?

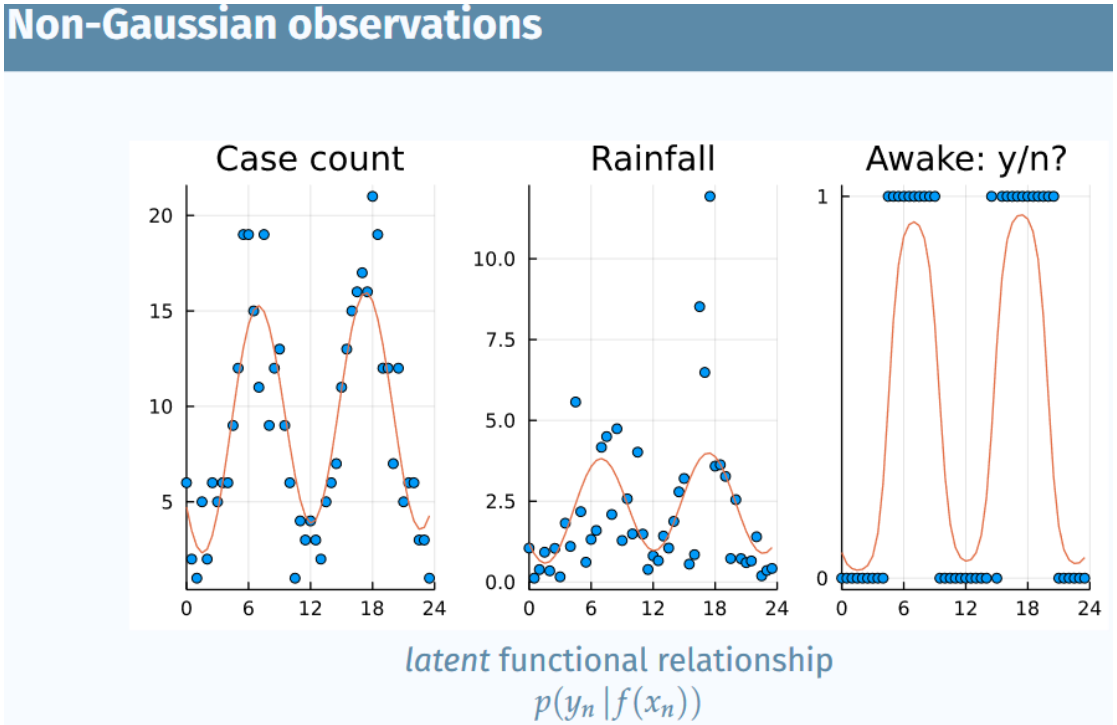
- Monde Gaussien: symmétrique, pas d'outliers, valeurs réelles

$$\overbrace{p(f_\theta|y,\mathbf{x},x_0)}^{posterior=Gaussien} = \frac{\overbrace{p(y|\mathbf{x},f_\theta)}^{likelihood=Gaussien}\ \overbrace{p(f_\theta)}^{prior=Gaussien}}{\underbrace{p(y|\mathbf{x})}_{Gaussien}} = GP(m_*, k_{*,\theta})$$

Limites

GP pour les données non-Gaussiennes ?

- Mondes *non-Gaussiens*: valeurs positives (durées, pluviométrie), comptage (nb d'accidents), valeurs extrèmes (rendements financiers), taux, catégorielles, etc.



Source: Gaussian Process Summer School 22'

Limites

GP pour les données non-Gaussiennes ?

- Mondes *non-Gaussiens*: valeurs positives (durées, pluviométrie), comptage (nb d'accidents), valeurs extrèmes (rendements financiers), taux, catégorielles, etc.
- Principe: transformer $\mathbb{E}(y)$ (espace latent) puis faire une hypothèse de GP sur la résultante. Prédiction en opération inverse.

Limites

GP pour les données non-Gaussiennes ?

- Mondes *non-Gaussiens*: valeurs positives (durées, pluviométrie), comptage (nb d'accidents), valeurs extrèmes (rendements financiers), taux, catégorielles, etc.
- Principe: transformer $\mathbb{E}(y)$ (espace latent) puis faire une hypothèse de GP sur la résultante. Prédiction en opération inverse.
- Exempe (classification): modéliser inv-logit$[p(y = \text{Awake})]$ avec un *GP*

Limites

GP pour les données non-Gaussiennes ?

- Mondes *non-Gaussiens*: valeurs positives (durées, pluviométrie), comptage (nb d'accidents), valeurs extrèmes (rendements financiers), taux, catégorielles, etc.
- Principe: transformer $\mathbb{E}(y)$ (espace latent) puis faire une hypothèse de GP sur la résultante. Prédiction en opération inverse.
- Exempe (classification): modéliser inv-logit$[p(y = \text{Awake})]$ avec un *GP*
- Conséquence:

Limites

GP pour les données non-Gaussiennes ?

- Mondes *non-Gaussiens*: valeurs positives (durées, pluviométrie), comptage (nb d'accidents), valeurs extrèmes (rendements financiers), taux, catégorielles, etc.
- Principe: transformer $\mathbb{E}(y)$ (espace latent) puis faire une hypothèse de GP sur la résultante. Prédiction en opération inverse.
- Exempe (classification): modéliser inv-logit$[p(y = \text{Awake})]$ avec un *GP*
- Conséquence:

$$\overbrace{p(f_\theta|y,\mathbf{x},x_0)}^{posterior=non-Gaussien} = \frac{\overbrace{p(y|\mathbf{x},f_\theta)}^{likelihood=non-Gaussien}\ \overbrace{p(f_\theta)}^{prior=Gaussien}}{\underbrace{p(y|\mathbf{x})}_{non-Gaussien}} = \cancel{GP(m_*,k_{*,\theta})}$$

Avantages / Limites des *GP* naïfs

Remèdes

$GP(m_*, k_{*,\theta})$ est incalculable analytiquement ou trop coûteuse,

- Méthodes variationnelles: Bruinsma **andothers** 2020; Hensman **andothers** 2015
- $GP(m_*, k_{*,\theta})$ est *approximée* par une distribution "variationnelle" q_ψ, plus simple à apprendre

[7] Ex: divergence KL.

Remèdes

$GP(m_*, k_{*,\theta})$ est incalculable analytiquement ou trop coûteuse,

- Méthodes variationnelles: Bruinsma **andothers** 2020; Hensman **andothers** 2015
- $GP(m_*, k_{*,\theta})$ est *approximée* par une distribution "variationnelle" q_ψ, plus simple à apprendre
- (hyper-) paramètres appris par minimisation d'une "différence"[7] entre q_ψ et $GP(m_*, k_{*,\theta})$

[7] Ex: divergence KL.

Remèdes

$GP(m_*, k_{*,\theta})$ est incalculable analytiquement ou trop coûteuse,

- Méthodes variationnelles: Bruinsma **andothers** 2020; Hensman **andothers** 2015
- $GP(m_*, k_{*,\theta})$ est *approximée* par une distribution "variationnelle" q_ψ, plus simple à apprendre
- (hyper-) paramètres appris par minimisation d'une "différence"[7] entre q_ψ et $GP(m_*, k_{*,\theta})$
- Le modèle d'inférence est donc appris en résolvant un problème d'*optimisation*: gradient stochastique, differentiation automatique (PyTorch, Tensorflow, JAX), etc.

[7] Ex: divergence KL.

Remèdes

Méthodes d'*échantillonnage* (MCMC, importance sampling, MH, etc): Rasmussen **and** Williams 2006

- Principe: remplacer $GP(m_*, k_{*,\theta})$ par un échantillonneur peu coutêux

Remèdes

Méthodes d'*échantillonnage* (MCMC, importance sampling, MH, etc): Rasmussen **and** Williams 2006

- Principe: remplacer $GP(m_*, k_{*,\theta})$ par un échantillonneur peu coutêux
- $m_*, k_{*,\theta}$ restent inconnus et sont estimés en échantillonnant (beaucoup) puis en moyennant (loi forte des grands nombres)

Remèdes

Méthodes d'*échantillonnage* (MCMC, importance sampling, MH, etc): Rasmussen **and** Williams 2006

- Principe: remplacer $GP(m_*, k_{*,\theta})$ par un échantillonneur peu coutêux
- $m_*, k_{*,\theta}$ restent inconnus et sont estimés en échantillonnant (beaucoup) puis en moyennant (loi forte des grands nombres)

Méthode de *Kalman* ("State-space" model): Solin **and** Särkkä 2014; Titsias **andothers** 2024

- (uniquement pour les données temporelles)

Remèdes

Méthodes d'*échantillonnage* (MCMC, importance sampling, MH, etc): Rasmussen **and** Williams 2006

- Principe: remplacer $GP(m_*, k_{*,\theta})$ par un échantillonneur peu coutêux
- $m_*, k_{*,\theta}$ restent inconnus et sont estimés en échantillonnant (beaucoup) puis en moyennant (loi forte des grands nombres)

Méthode de *Kalman* ("State-space" model): Solin **and** Särkkä 2014; Titsias **andothers** 2024

- (uniquement pour les données temporelles)
- Réécriture du *GP* en équation différentielle stochastique

Remèdes

Méthodes d'*échantillonnage* (MCMC, importance sampling, MH, etc): Rasmussen **and** Williams 2006

- Principe: remplacer $GP(m_*, k_{*,\theta})$ par un échantillonneur peu coutêux
- $m_*, k_{*,\theta}$ restent inconnus et sont estimés en échantillonnant (beaucoup) puis en moyennant (loi forte des grands nombres)

Méthode de *Kalman* ("State-space" model): Solin **and** Särkkä 2014; Titsias **andothers** 2024

- (uniquement pour les données temporelles)
- Réécriture du GP en équation différentielle stochastique
- Apprentissage par l'algorithme de Kalman: ~~$\mathcal{O}(n^3)$~~, calcul $\mathcal{O}(n)$

Remèdes

Méthodes d'*échantillonnage* (MCMC, importance sampling, MH, etc): Rasmussen **and** Williams 2006

- Principe: remplacer $GP(m_*, k_{*,\theta})$ par un échantillonneur peu coutêux
- $m_*, k_{*,\theta}$ restent inconnus et sont estimés en échantillonnant (beaucoup) puis en moyennant (loi forte des grands nombres)

Méthode de *Kalman* ("State-space" model): Solin **and** Särkkä 2014; Titsias **andothers** 2024

- (uniquement pour les données temporelles)
- Réécriture du GP en équation différentielle stochastique
- Apprentissage par l'algorithme de Kalman: ~~$\mathcal{O}(n^3)$~~, calcul $\mathcal{O}(n)$
- La réecriture dépend du kernel: nécessite un kernel "pas trop compliqué" (état de recherche)

Merci pour votre attention.
Questions ?

Références I

- Borovitskiy, Viacheslav **andothers** (2021). "Matérn Gaussian Processes on Graphs". **in**.
- Bruinsma, Wessel P **andothers** (2020). "Scalable Exact Inference in Multi-Output Gaussian Processes". **in**.
- Hensman, James **andothers** (2015). "Scalable Variational Gaussian Process Classification". **in**.
- Parra, Gabriel **and** Felipe Tobar (2017). "Spectral mixture kernels for multi-output Gaussian processes". **in**.
- Rasmussen, Carl Edward **and** Christopher K.I Williams (2006). *Gaussian Processes for Machine Learning*. MIT Press. ISBN: 026218253X.
- Solin, Arno **and** Simo Särkkä (2014). "Explicit Link Between Periodic Covariance Functions and State Space Models". **in**.
- Titsias, Michalis K **andothers** (2024). "Kalman Filter for Online Classification of Non-Stationary Data". **in**.

Références II

Wilson, Andrew Gordon **andothers** (2016). “Deep Kernel Learning”. **in**.