Using GP emulation in cardiovascular modelling

Mihaela Paun

Joint work with: Sean McGinty, André Fensterseifer Schmidt, Mitchel Colebank, Dirk Husmeier

School of Mathematics and Statistics
University of Glasgow

September, 2024

Table of contents

Overview of applications

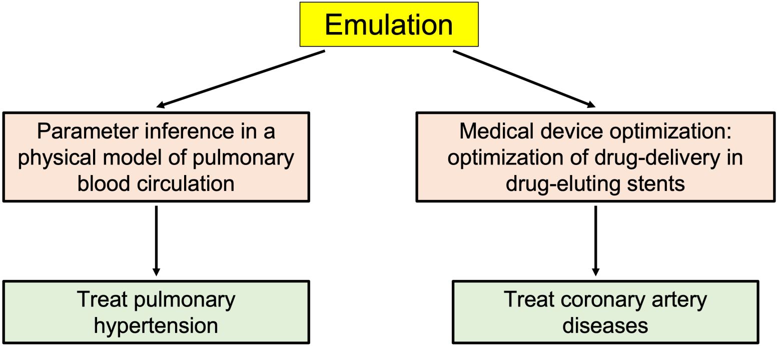
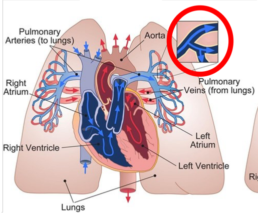


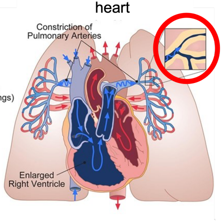
Table of contents

Background

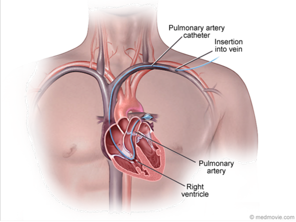
Healthy heart



Pulmonary hypertension heart

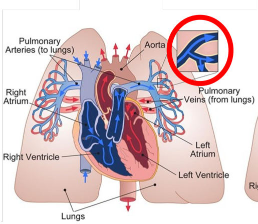


Pulmonary hypertension diagnosis: invasive right-heart catheterisation

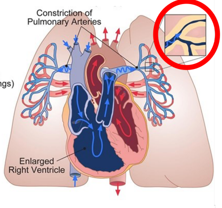


Background

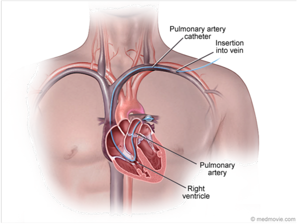
Healthy heart



Pulmonary hypertension heart



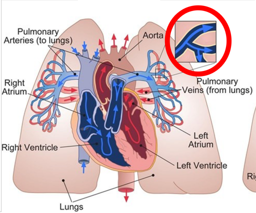
Pulmonary hypertension diagnosis: invasive right-heart catheterisation



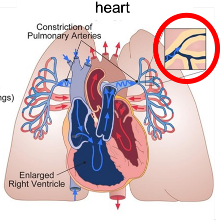
- Pulmonary hypertension (PH): high blood pressure in the pulmonary arteries, which are stiff and thick

Background

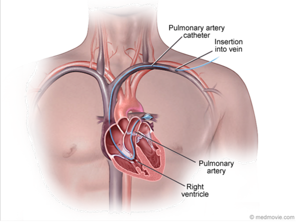
Healthy heart



Pulmonary hypertension heart



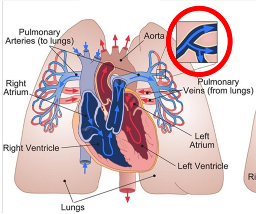
Pulmonary hypertension diagnosis: invasive right-heart catheterisation



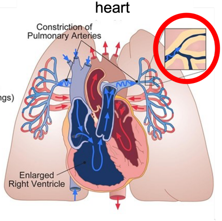
- Pulmonary hypertension (PH): high blood pressure in the pulmonary arteries, which are stiff and thick
- If PH left untreated → right-heart damage, heart failure

Background

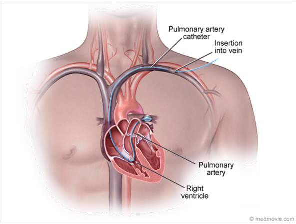
Healthy heart



Pulmonary hypertension heart

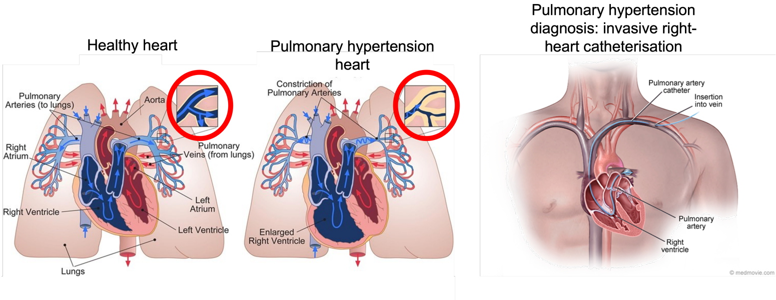


Pulmonary hypertension diagnosis: invasive right-heart catheterisation



- Pulmonary hypertension (PH): high blood pressure in the pulmonary arteries, which are stiff and thick
- If PH left untreated $\rightarrow$ right-heart damage, heart failure
- PH diagnosis: invasively measure pulmonary pressure with right-heart catheterisation $\rightarrow$ excessive bleeding, partial lung collapse

Background



- Pulmonary hypertension (PH): high blood pressure in the pulmonary arteries, which are stiff and thick
- If PH left untreated → right-heart damage, heart failure
- PH diagnosis: invasively measure pulmonary pressure with right-heart catheterisation → excessive bleeding, partial lung collapse
- Aim: Develop a non-invasive alternative (flow-based).

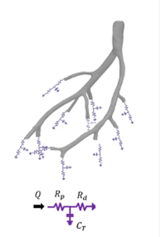
Pulmonary model

Parameters

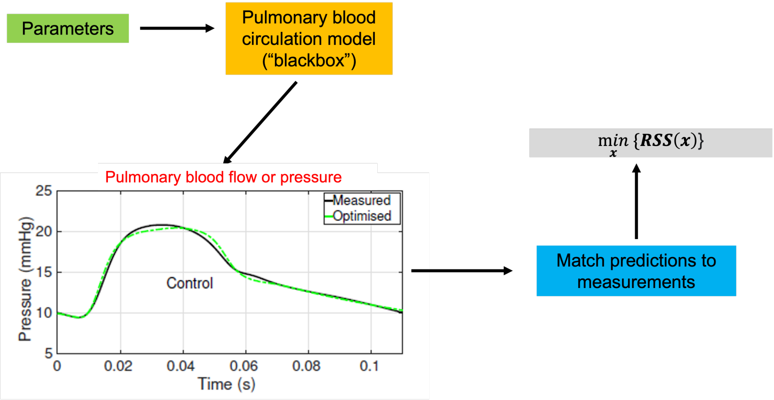
- Vessel stiffness
- (Windkessel) boundary conditions parameters

Pulmonary blood circulation model ("blackbox")

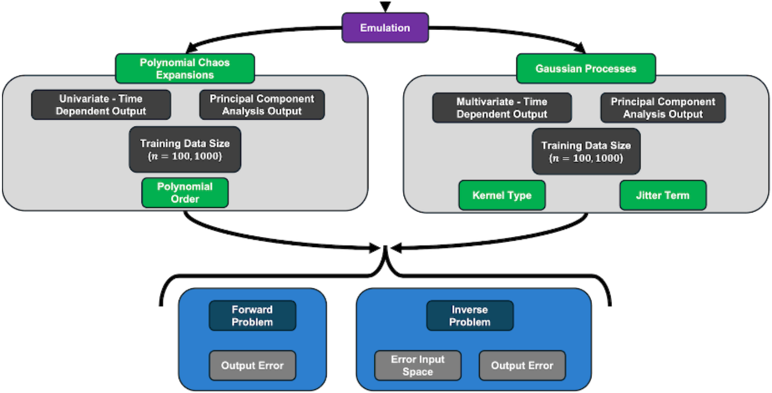
- Fluid-dynamics model for blood flow and pressure in pulmonary vessels



Parameter inference



Workflow


Emulation
Polynomial Chaos Expansions
Univariate - Time Dependent Output
Principal Component Analysis Output
Training Data Size ($n = 100, 1000$)
Polynomial Order
Gaussian Processes
Multivariate - Time Dependent Output
Principal Component Analysis Output
Training Data Size ($n = 100, 1000$)
Kernel Type
Jitter Term
Forward Problem
Output Error
Inverse Problem
Error Input Space
Output Error

Output representation

- Emulator in simulator output space (time series):

$$f(\boldsymbol{\theta}) = \mathbf{y} = (y_1, \ldots y_m), \tag{1}$$

Output representation

- Emulator in simulator output space (time series):

$$f(\boldsymbol{\theta}) = \mathbf{y} = (y_1, \ldots y_m), \tag{1}$$

- Emulator in PCA-reduced space:

$$f(\boldsymbol{\theta}) = \boldsymbol{\mu} + \sum_{j=1}^{q} c_j(\boldsymbol{\theta})\gamma_j + \epsilon(\boldsymbol{\theta}) \tag{2}$$

Output representation

- Emulator in simulator output space (time series):

$$f(\boldsymbol{\theta}) = \mathbf{y} = (y_1, \dots y_m), \tag{1}$$

- Emulator in PCA-reduced space:

$$f(\boldsymbol{\theta}) = \boldsymbol{\mu} + \sum_{j=1}^{q} c_j(\boldsymbol{\theta})\gamma_j + \epsilon(\boldsymbol{\theta}) \tag{2}$$

where $\boldsymbol{\mu}$: mean of training set; $\boldsymbol{\Gamma}_q = (\gamma_1, \dots, \gamma_q)$: basis; $c_j(\boldsymbol{\theta})$: coefficient (or PC score), $\epsilon(\boldsymbol{\theta})$: residual.

Emulator PCA

- Emulator in PCA-reduced space:

$$f(\boldsymbol{\theta}) = \boldsymbol{\mu} + \sum_{j=1}^{q} c_j(\boldsymbol{\theta})\boldsymbol{\gamma}_j + \epsilon(\boldsymbol{\theta}) \tag{3}$$

Emulator PCA

- Emulator in PCA-reduced space:

$$f(\boldsymbol{\theta}) = \boldsymbol{\mu} + \sum_{j=1}^{q} c_j(\boldsymbol{\theta})\gamma_j + \epsilon(\boldsymbol{\theta}) \quad (3)$$

- Fit independent GP emulators for each PC score:

$$c_j(\boldsymbol{\Theta})|\gamma \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}|\gamma), \quad j = 1, \ldots, q, \quad (4)$$

Emulator PCA

- Emulator in PCA-reduced space:

$$f(\boldsymbol{\theta}) = \boldsymbol{\mu} + \sum_{j=1}^{q} c_j(\boldsymbol{\theta})\gamma_j + \epsilon(\boldsymbol{\theta}) \tag{3}$$

- Fit independent GP emulators for each PC score:

$$c_j(\boldsymbol{\Theta})|\gamma \sim \mathrm{GP}(\mathbf{0}, \mathbf{K}|\gamma), \quad j = 1, \ldots, q, \tag{4}$$

where $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n)$: input set, $\mathbf{K} = [k(\boldsymbol{\theta}_l, \boldsymbol{\theta}_p)]_{l,p=1}^{n}$: covariance matrix, $k(\cdot)$: kernel

Emulator time series

- Emulator in simulator output space (time series):

$$f(\boldsymbol{\theta}) = \mathbf{y} = (y_1, \dots y_m), \tag{5}$$

Emulator time series

- Emulator in simulator output space (time series):

$$f(\boldsymbol{\theta}) = \mathbf{y} = (y_1, \ldots y_m), \tag{5}$$

- GP input: $(\boldsymbol{\theta}, t) \rightarrow$ *univariate* output: $f(\boldsymbol{\theta}, t) = y_t$

$$f(\boldsymbol{\Theta}_{\boldsymbol{\theta},t})|\tilde{\gamma} \sim \mathrm{GP}(\mathbf{0}, \tilde{\mathbf{K}}|\tilde{\gamma}), \tag{6}$$

Emulator time series

- Emulator in simulator output space (time series):

$$f(\boldsymbol{\theta}) = \mathbf{y} = (y_1, \ldots y_m), \tag{5}$$

- GP input: $(\boldsymbol{\theta}, t) \rightarrow$ *univariate* output: $f(\boldsymbol{\theta}, t) = y_t$

$$f(\boldsymbol{\Theta}_{\boldsymbol{\theta},t})|\tilde{\gamma} \sim \mathrm{GP}(\mathbf{0}, \tilde{\mathbf{K}}|\tilde{\gamma}), \tag{6}$$

- Assume separability in kernels between inputs $\boldsymbol{\theta}$ and t:

$$k((t_i, \boldsymbol{\theta}_i), (t_j, \boldsymbol{\theta}_j)) = k_t(t_i, t_j) k_{\boldsymbol{\theta}}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j), \tag{7}$$

Emulator time series

- Emulator in simulator output space (time series):

$$f(\boldsymbol{\theta}) = \mathbf{y} = (y_1, \dots y_m), \tag{5}$$

- GP input: $(\boldsymbol{\theta}, t) \rightarrow$ *univariate* output: $f(\boldsymbol{\theta}, t) = y_t$

$$f(\boldsymbol{\Theta}_{\boldsymbol{\theta},t})|\tilde{\gamma} \sim \text{GP}(\mathbf{0}, \tilde{\mathbf{K}}|\tilde{\gamma}), \tag{6}$$

- Assume separability in kernels between inputs $\boldsymbol{\theta}$ and t:

$$k((t_i, \boldsymbol{\theta}_i), (t_j, \boldsymbol{\theta}_j)) = k_t(t_i, t_j) k_{\boldsymbol{\theta}}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j), \tag{7}$$

- Represent full covariance matrix as the Kronecker product between two smaller matrices:

$$\tilde{\mathbf{K}}(\boldsymbol{\Theta}_{\theta,t}, \boldsymbol{\Theta}_{\theta,t}) = \mathbf{K}_t(\mathbf{t}, \mathbf{t}) \otimes \mathbf{K}_{\boldsymbol{\theta}}(\boldsymbol{\Theta}, \boldsymbol{\Theta}). \tag{8}$$

PCE

- PCE emulators live in a polynomial function space.

PCE

- PCE emulators live in a polynomial function space.
- PCE approximates the simulator by finite truncation

$$f(\boldsymbol{\theta}) = \sum_{j=0}^{\mathcal{J}-1} z_j \Psi_j(\boldsymbol{\theta}), \quad \Psi_j(\boldsymbol{\theta}) = \prod_{i=1}^{d} \psi_{ij}(\theta_i) \qquad (9)$$

PCE

- PCE emulators live in a polynomial function space.
- PCE approximates the simulator by finite truncation

$$f(\boldsymbol{\theta}) = \sum_{j=0}^{\mathcal{J}-1} z_j \Psi_j(\boldsymbol{\theta}), \quad \Psi_j(\boldsymbol{\theta}) = \prod_{i=1}^{d} \psi_{ij}(\theta_i) \tag{9}$$

where z_j: polynomial coefficients corresponding to a specific family of polynomials; $\Psi_j(\boldsymbol{\theta})$: multivariate polynomials for $\boldsymbol{\theta} = (\theta_1, \ldots \theta_d)$, constructed from a product of univariate polynomials $\psi_{ij}(\theta_i)$; $\mathcal{J} = \binom{d+\mathcal{K}}{\mathcal{K}}$: total number of polynomial basis functions for polynomial order of $\mathcal{K}$.

PCE

- PCE emulators live in a polynomial function space.
- PCE approximates the simulator by finite truncation

$$f(\boldsymbol{\theta}) = \sum_{j=0}^{\mathcal{J}-1} z_j \Psi_j(\boldsymbol{\theta}), \quad \Psi_j(\boldsymbol{\theta}) = \prod_{i=1}^{d} \psi_{ij}(\theta_i) \qquad (9)$$

where z_j: polynomial coefficients corresponding to a specific family of polynomials; $\Psi_j(\boldsymbol{\theta})$: multivariate polynomials for $\boldsymbol{\theta} = (\theta_1, \ldots \theta_d)$, constructed from a product of univariate polynomials $\psi_{ij}(\theta_i)$; $\mathcal{J} = \binom{d+\mathcal{K}}{\mathcal{K}}$: total number of polynomial basis functions for polynomial order of $\mathcal{K}$.

- Fit independent PCEs for each output time point: $f(\boldsymbol{\theta}, t) = \sum_{j=0}^{\mathcal{J}-1} z_{jt} \Psi_j(\boldsymbol{\theta})$.

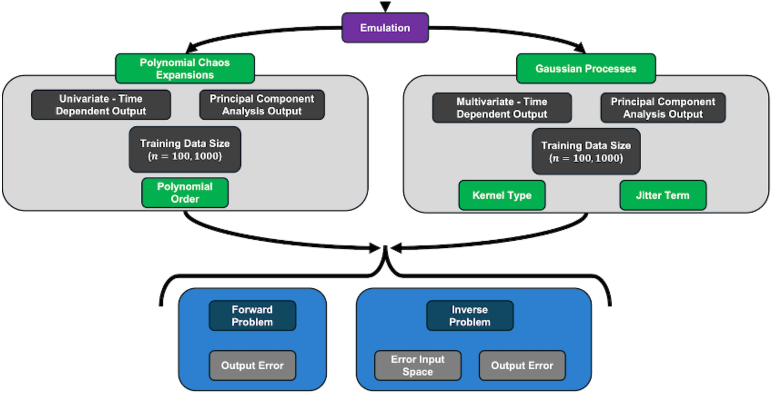
PCE

- PCE emulators live in a polynomial function space.
- PCE approximates the simulator by finite truncation

$$f(\boldsymbol{\theta}) = \sum_{j=0}^{\mathcal{J}-1} z_j \Psi_j(\boldsymbol{\theta}), \quad \Psi_j(\boldsymbol{\theta}) = \prod_{i=1}^{d} \psi_{ij}(\theta_i) \tag{9}$$

  where z_j: polynomial coefficients corresponding to a specific family of polynomials; $\Psi_j(\boldsymbol{\theta})$: multivariate polynomials for $\boldsymbol{\theta} = (\theta_1, \ldots \theta_d)$, constructed from a product of univariate polynomials $\psi_{ij}(\theta_i)$; $\mathcal{J} = \binom{d+\mathcal{K}}{\mathcal{K}}$: total number of polynomial basis functions for polynomial order of $\mathcal{K}$.
- Fit independent PCEs for each output time point: $f(\boldsymbol{\theta}, t) = \sum_{j=0}^{\mathcal{J}-1} z_{jt} \Psi_j(\boldsymbol{\theta})$.
- Fit independent PCEs for each PCA score: $c_k(\boldsymbol{\theta}) = \sum_{j=0}^{\mathcal{J}-1} z_{jk} \Psi_j(\boldsymbol{\theta})$.

Workflow


Emulation
Polynomial Chaos Expansions
Univariate - Time Dependent Output
Principal Component Analysis Output
Training Data Size ($n = 100, 1000$)
Polynomial Order
Gaussian Processes
Multivariate - Time Dependent Output
Principal Component Analysis Output
Training Data Size ($n = 100, 1000$)
Kernel Type
Jitter Term
Forward Problem
Output Error
Inverse Problem
Error Input Space
Output Error

Forward problem

- Investigate (1) effect of GP kernel type, PCE polynomial order, and training size on predictive performance; (2) time versus PCA representation; (3) PCE versus GP

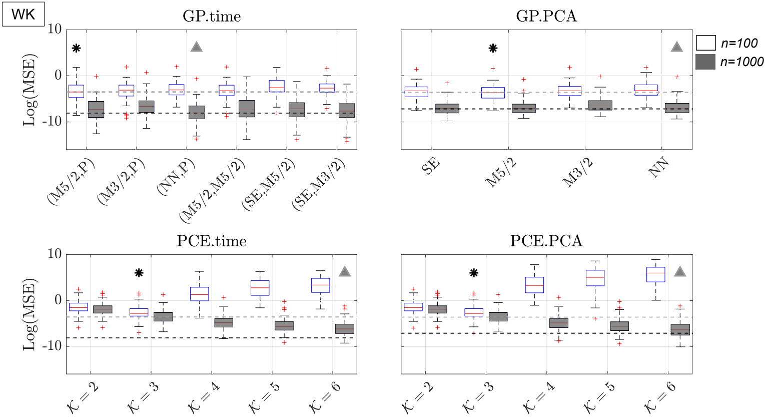
Forward problem

- Investigate (1) effect of GP kernel type, PCE polynomial order, and training size on predictive performance; (2) time versus PCA representation; (3) PCE versus GP
- Error in output space:

$$\mathrm{MSE}(\boldsymbol{\theta}_j^{\text{test}}) = \frac{1}{m}\sum_{i=1}^{m}\left(y_i - \mathcal{M}(\boldsymbol{\theta}_j^{\text{test}}, t_i)\right)^2, \qquad (10)$$

where $\mathcal{M}(.)$: emulator (GP/PCE) prediction

Results - forward problem



Best methods: GP-time and GP-PCA with 1000 training points.

Inverse problem

- Gradient-based optimisation using the emulators on simulated and noise-free data

Inverse problem

- Gradient-based optimisation using the emulators on simulated and noise-free data
- Output error:

$$\mathrm{MSE}(\hat{\boldsymbol{\theta}}_j) = \frac{1}{m}\sum_{i=1}^{m}\left(y_i - f(\hat{\boldsymbol{\theta}}_j, t_i)\right)^2, \qquad (11)$$

where $\hat{\boldsymbol{\theta}}_j$: inferred parameter vector for j^{th} test data set, $f(.)$: simulator output.

Inverse problem

- Gradient-based optimisation using the emulators on simulated and noise-free data
- Output error:

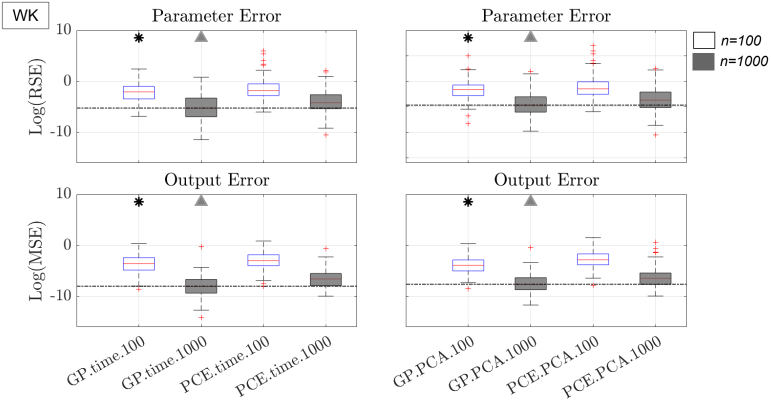
$$\mathrm{MSE}(\hat{\boldsymbol{\theta}}_j) = \frac{1}{m}\sum_{i=1}^{m}\left(y_i - f(\hat{\boldsymbol{\theta}}_j, t_i)\right)^2, \qquad (11)$$

where $\hat{\boldsymbol{\theta}}_j$: inferred parameter vector for j^{th} test data set, $f(.)$: simulator output.

- Input (parameter) error:

$$\mathrm{RSE}(\hat{\boldsymbol{\theta}}_j) = \sum_{l=1}^{d}\left(\frac{\theta_{j,l}^{\text{test}} - \hat{\theta}_{j,l}}{\theta_{j,l}^{\text{test}}}\right)^2. \qquad (12)$$

Results - inverse problem



Best methods: GP-time and GP-PCA with 1000 training points.

Final remarks

- We have constructed surrogates models for the pulmonary blood pressure with GPs and PCEs for two output representations: time series and PCA.

Final remarks

- We have constructed surrogates models for the pulmonary blood pressure with GPs and PCEs for two output representations: time series and PCA.
- Forward problem: we have assessed the effect of different settings (GP kernel, PCE polynomial order, training size) on output prediction.

Final remarks

- We have constructed surrogates models for the pulmonary blood pressure with GPs and PCEs for two output representations: time series and PCA.
- Forward problem: we have assessed the effect of different settings (GP kernel, PCE polynomial order, training size) on output prediction.
- We have taken forward the best settings w.r.t. the forward problem and assessed inference accuracy.

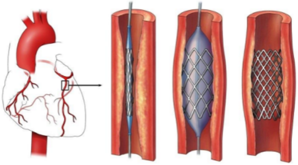
Final remarks

- We have constructed surrogates models for the pulmonary blood pressure with GPs and PCEs for two output representations: time series and PCA.
- Forward problem: we have assessed the effect of different settings (GP kernel, PCE polynomial order, training size) on output prediction.
- We have taken forward the best settings w.r.t. the forward problem and assessed inference accuracy.
- Finding: best methods are GP-time and GP-PCA with 1000 training points for forward and inverse problems.

Table of contents

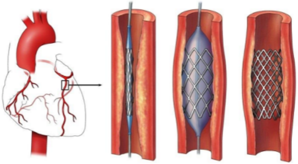
Background

- Stent implantation with antiproliferative drugs treats obstructive coronary artery disease



Background

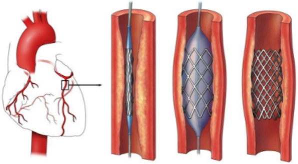
- Stent implantation with antiproliferative drugs treats obstructive coronary artery disease



- Safety: maintain drug levels below a toxic level

Background

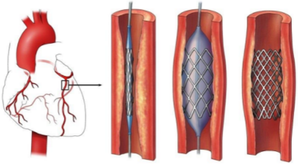
- Stent implantation with antiproliferative drugs treats obstructive coronary artery disease



- Safety: maintain drug levels below a toxic level
- Efficacy: saturate with drug receptors target cells in arterial wall long enough

Background

- Stent implantation with antiproliferative drugs treats obstructive coronary artery disease

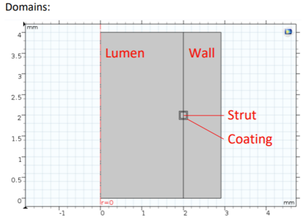


- Safety: maintain drug levels below a toxic level
- Efficacy: saturate with drug receptors target cells in arterial wall long enough
- Aim: find optimum stent design parameters to balance safety and efficacy

Stents model

Stents design parameters → Stents model ("blackbox")

- Initial drug mass
- Polymer coating thickness
- Drug diffusion coefficient



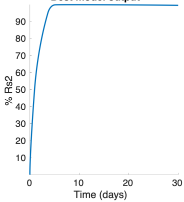
(2D axisymmetric geometry representing a blood vessel)

- Fluid-dynamics model for the blood flow in the lumen (Navier Stokes equations)
- Fluid flow across the (pourous) arterial wall (Darcy's law)
- Drug transport: drug release from the stent coating => uptake by the arterial wall (therapeutic aspect) and exhaust via blood flow (efficiency loss and possible toxic effect) ; advection-diffusion-reaction equation

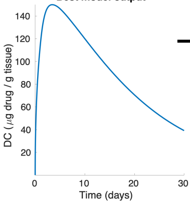
Stents optimisation

Stents design parameters → Stents model ("blackbox")

Drug receptor saturation (Rs2)



Drug content (DC)



Maximise Rs2 (efficacy) s.t. max(DC)-150 ≤ 0 (safety)

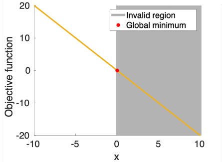
$\min_{\boldsymbol{x}} \{f(\boldsymbol{x}) : \boldsymbol{c}(\boldsymbol{x}) \leq \boldsymbol{0}\}$

$f(\boldsymbol{x})$ defined for both $\boldsymbol{c}(\boldsymbol{x}) \leq \boldsymbol{0}$ ("safe" region) & $\boldsymbol{c}(\boldsymbol{x}) > \boldsymbol{0}$ ("critical" region)

Challenge

Constrained global optimum on the constraint boundary

1D benchmark illustration



Conventional Bayesian optimisation

- Bayesian optimisation (BO): global method suitable for computationally expensive OFs

Conventional Bayesian optimisation

- Bayesian optimisation (BO): global method suitable for computationally expensive OFs
- Conventional BO is unconstrained

Conventional Bayesian optimisation

- Bayesian optimisation (BO): global method suitable for computationally expensive OFs
- Conventional BO is unconstrained
- BO builds a surrogate model of $f(\mathbf{x})$ (with Gaussian Processes, GPs)

Acquisition functions

- BO maximises a computationally cheap acquisition function (AF) by balancing exploration (surrogate uncertainty) and exploitation (low surrogate values)

Acquisition functions

- BO maximises a computationally cheap acquisition function (AF) by balancing exploration (surrogate uncertainty) and exploitation (low surrogate values)
- Upper confidence bound (UCB):

$$\alpha_{\mathrm{UCB}}(\mathbf{x}) = -m(\mathbf{x}) + \beta\sigma(\mathbf{x})$$

where $m(.)$, $\sigma(.)$: GP posterior predictive mean & standard deviation

Acquisition functions

- BO maximises a computationally cheap acquisition function (AF) by balancing exploration (surrogate uncertainty) and exploitation (low surrogate values)
- Upper confidence bound (UCB):

$$\alpha_{\mathrm{UCB}}(\mathbf{x}) = -m(\mathbf{x}) + \beta\sigma(\mathbf{x})$$

where $m(.)$, $\sigma(.)$: GP posterior predictive mean & standard deviation

- Expected improvement (EI):

$$\alpha_{\mathrm{EI}}(\mathbf{x}) = (f_{\min} - m(\mathbf{x}))\Phi\left(\frac{f_{\min} - m(\mathbf{x})}{\sigma(\mathbf{x})}\right) + \sigma(\mathbf{x})\phi\left(\frac{f_{\min} - m(\mathbf{x})}{\sigma(\mathbf{x})}\right)$$

Bayesian optimisation - Illustration

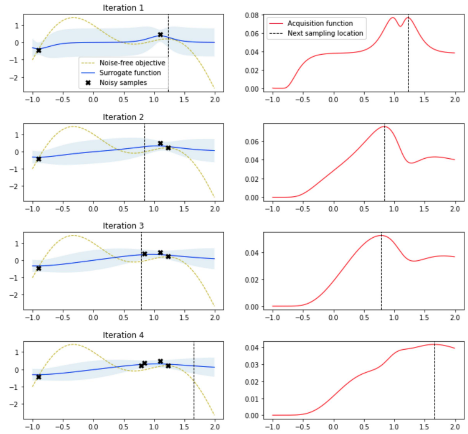
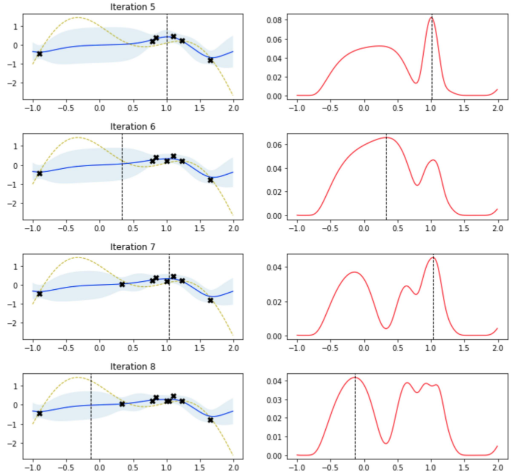
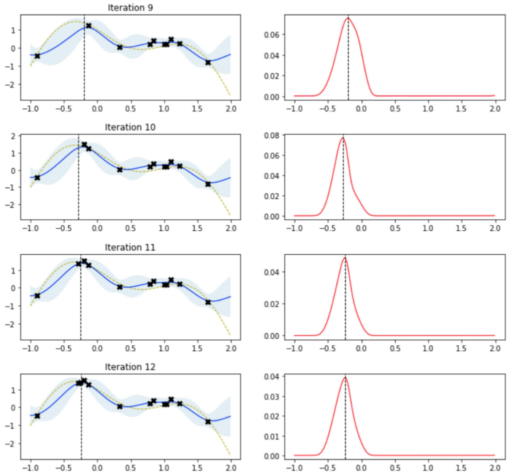


Figure: Source: `https://medium.com/analytics-vidhya/bayesian-optimization-9ddb3aff0eb4`

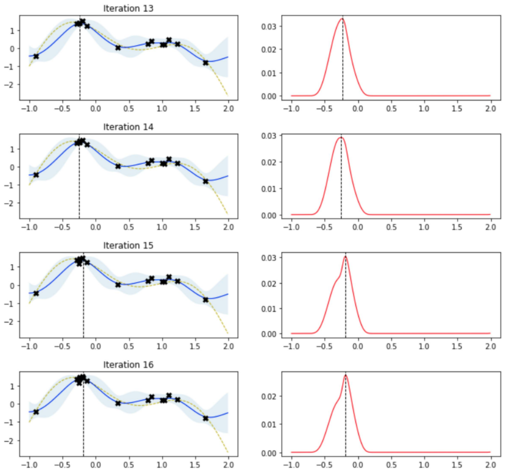
Bayesian optimisation - Illustration



Bayesian optimisation - Illustration



Bayesian optimisation - Illustration



Constrained BO

- Learn constraint function with GP classifier or regression

Constrained BO

- Learn constraint function with GP classifier or regression
- GP-classifier based methods use predicted probability of constraint satisfaction:

Constrained BO

- Learn constraint function with GP classifier or regression
- GP-classifier based methods use predicted probability of constraint satisfaction:
  - Constrained (C) $\rightarrow$ CEI, CUCB

Constrained BO

- Learn constraint function with GP classifier or regression
- GP-classifier based methods use predicted probability of constraint satisfaction:
  - Constrained (C) → CEI, CUCB
  - Asymmetric entropy (AE) → EI-AE, UCB-AE

Constrained BO

- Learn constraint function with GP classifier or regression
- GP-classifier based methods use predicted probability of constraint satisfaction:
  - Constrained (C) $\rightarrow$ CEI, CUCB
  - Asymmetric entropy (AE) $\rightarrow$ EI-AE, UCB-AE
- GP-regression based methods enforce a penalty in the critical input domain:
  - Augmented Lagrangian (AL) $\rightarrow$ EI-AL, UCB-AL

Constrained BO

- Learn constraint function with GP classifier or regression
- GP-classifier based methods use predicted probability of constraint satisfaction:
  - Constrained (C) → CEI, CUCB
  - Asymmetric entropy (AE) → EI-AE, UCB-AE
- GP-regression based methods enforce a penalty in the critical input domain:
  - Augmented Lagrangian (AL) → EI-AL, UCB-AL
  - Barrier method (BM) → EI-BM, UCB-BM, Mean-BM

Constrained BO

- Learn constraint function with GP classifier or regression
- GP-classifier based methods use predicted probability of constraint satisfaction:
  - Constrained (C) → CEI, CUCB
  - Asymmetric entropy (AE) → EI-AE, UCB-AE
- GP-regression based methods enforce a penalty in the critical input domain:
  - Augmented Lagrangian (AL) → EI-AL, UCB-AL
  - Barrier method (BM) → EI-BM, UCB-BM, Mean-BM

	Acquisition function		
Method	EI	UCB	Mean
C	*CEI*	*CUCB*	-
AE	*EI-AE*	*UCB-AE*	-
AL	*EI-AL*	*UCB-AL*	-
BM	*EI-BM*	*UCB-BM*	*Mean-BM*

Constrained BO: GP-classifier based methods

- Constrained (C) $\rightarrow$ CEI, CUCB:

Constrained BO: GP-classifier based methods

- Constrained (C) $\rightarrow$ CEI, CUCB:

$$\alpha_{\text{CEI/CUCB}}(\mathbf{x}) = \alpha_{\text{EI/UCB}}(\mathbf{x}) \prod_{j=1}^{m} p(c_j(\mathbf{x}) \leq 0)$$

where $p(\mathbf{c}(\mathbf{x}) \leq \mathbf{0})$: predicted probability of constraint satisfaction.

Constrained BO: GP-classifier based methods

- Constrained (C) $\rightarrow$ CEI, CUCB:

$$\alpha_{\mathrm{CEI/CUCB}}(\mathbf{x}) = \alpha_{\mathrm{EI/UCB}}(\mathbf{x}) \prod_{j=1}^{m} p(c_j(\mathbf{x}) \leq 0)$$

where $p(\mathbf{c}(\mathbf{x}) \leq \mathbf{0})$: predicted probability of constraint satisfaction.

- Asymmetric entropy (AE) $\rightarrow$ EI-AE, UCB-AE:

Constrained BO: GP-classifier based methods

- Constrained (C) → CEI, CUCB:

$$\alpha_{\mathrm{CEI/CUCB}}(\mathbf{x}) = \alpha_{\mathrm{EI/UCB}}(\mathbf{x}) \prod_{j=1}^{m} p(c_j(\mathbf{x}) \leq 0)$$

where $p(\mathbf{c}(\mathbf{x}) \leq \mathbf{0})$: predicted probability of constraint satisfaction.

- Asymmetric entropy (AE) → EI-AE, UCB-AE:

$$\alpha_{\mathrm{EI\text{-}AE/UCB\text{-}AE}}(\mathbf{x}) = \alpha^{\omega_1}_{\mathrm{EI/UCB}}(\mathbf{x}) S_a^{\omega_2}(\mathbf{x})$$

Constrained BO: GP-classifier based methods

- Constrained (C) $\rightarrow$ CEI, CUCB:

$$\alpha_{\mathrm{CEI/CUCB}}(\mathbf{x}) = \alpha_{\mathrm{EI/UCB}}(\mathbf{x}) \prod_{j=1}^{m} p(c_j(\mathbf{x}) \leq 0)$$

where $p(\mathbf{c}(\mathbf{x}) \leq \mathbf{0})$: predicted probability of constraint satisfaction.

- Asymmetric entropy (AE) $\rightarrow$ EI-AE, UCB-AE:

$$\alpha_{\mathrm{EI\text{-}AE/UCB\text{-}AE}}(\mathbf{x}) = \alpha^{\omega_1}_{\mathrm{EI/UCB}}(\mathbf{x}) S_a^{\omega_2}(\mathbf{x})$$

$$S_a(\mathbf{x}) = \frac{2\prod_{j=1}^{m} p(c_j(\mathbf{x}) \leq 0)(1 - \prod_{j=1}^{m} p(c_j(\mathbf{x}) \leq 0))}{\prod_{j=1}^{m} p(c_j(\mathbf{x}) \leq 0) - 2w\prod_{j=1}^{m} p(c_j(\mathbf{x}) \leq 0) + w^2}$$

where $w = 2/3, \omega_1 = 1, \omega_2 = 5$.

Constrained BO: GP-regression based methods

- Augmented Lagrangian (AL) $\rightarrow$ EI-AL, UCB-AL:

Constrained BO: GP-regression based methods

- Augmented Lagrangian (AL) $\rightarrow$ EI-AL, UCB-AL:

$$L_{\mathrm{A}}(\mathbf{x};\boldsymbol{\lambda},\rho) = f(\mathbf{x}) + \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{c}(\mathbf{x}) + \frac{1}{2\rho}\sum_{j=1}^{m} c_j(\mathbf{x})^2$$

with tuning parameters ρ: penalty, $\boldsymbol{\lambda}$: Lagrange multipliers.

Constrained BO: GP-regression based methods

- Augmented Lagrangian (AL) $\rightarrow$ EI-AL, UCB-AL:

$$L_{\mathrm{A}}(\mathbf{x};\boldsymbol{\lambda},\rho) = f(\mathbf{x}) + \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{c}(\mathbf{x}) + \frac{1}{2\rho}\sum_{j=1}^{m} c_j(\mathbf{x})^2$$

with tuning parameters ρ: penalty, $\boldsymbol{\lambda}$: Lagrange multipliers.

$$Y(\mathbf{x}) = Y_f(\mathbf{x}) + \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{Y}_c(\mathbf{x}) + \frac{1}{2\rho}\sum_{j=1}^{m}(Y_{c_j}(\mathbf{x}))^2$$

Constrained BO: GP-regression based methods

- Augmented Lagrangian (AL) $\rightarrow$ EI-AL, UCB-AL:

$$L_{\mathrm{A}}(\mathbf{x}; \boldsymbol{\lambda}, \rho) = f(\mathbf{x}) + \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{c}(\mathbf{x}) + \frac{1}{2\rho}\sum_{j=1}^{m} c_j(\mathbf{x})^2$$

with tuning parameters ρ: penalty, $\boldsymbol{\lambda}$: Lagrange multipliers.

$$Y(\mathbf{x}) = Y_f(\mathbf{x}) + \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{Y}_c(\mathbf{x}) + \frac{1}{2\rho}\sum_{j=1}^{m}(Y_{c_j}(\mathbf{x}))^2$$

$$\alpha_{\text{EI-AL}}(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T}\max(0, y_{\min} - y^t(\mathbf{x})), \text{ via Monte Carlo}$$

Constrained BO: GP-regression based methods

- Augmented Lagrangian (AL) $\rightarrow$ EI-AL, UCB-AL:

$$L_{\mathrm{A}}(\mathbf{x}; \boldsymbol{\lambda}, \rho) = f(\mathbf{x}) + \boldsymbol{\lambda}^{\mathrm{T}} \mathbf{c}(\mathbf{x}) + \frac{1}{2\rho} \sum_{j=1}^{m} c_j(\mathbf{x})^2$$

with tuning parameters ρ: penalty, $\boldsymbol{\lambda}$: Lagrange multipliers.

$$Y(\mathbf{x}) = Y_f(\mathbf{x}) + \boldsymbol{\lambda}^{\mathrm{T}} \mathbf{Y}_c(\mathbf{x}) + \frac{1}{2\rho} \sum_{j=1}^{m} (Y_{c_j}(\mathbf{x}))^2$$

$$\alpha_{\text{EI-AL}}(\mathbf{x}) = \frac{1}{T} \sum_{t=1}^{T} \max(0, y_{\min} - y^t(\mathbf{x})), \text{ via Monte Carlo}$$

$$\alpha_{\text{UCB-AL}} = -m_Y(\mathbf{x}) + \beta \sigma_Y(\mathbf{x}), \text{ analytical form}$$

Constrained BO: GP-regression based methods

- Barrier method (BM) $\rightarrow$ EI-BM, UCB-BM, Mean-BM:

Constrained BO: GP-regression based methods

- Barrier method (BM) $\rightarrow$ EI-BM, UCB-BM, Mean-BM:

$$B(\mathbf{x};\gamma) = f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\Bigg(\log\Bigg(\max\Bigg(-c_j(\mathbf{x}), 10^{-10}\Bigg)\Bigg)\Bigg)$$

Constrained BO: GP-regression based methods

- Barrier method (BM) → EI-BM, UCB-BM, Mean-BM:

$$B(\mathbf{x};\gamma) = f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\Bigg(\log\Bigg(\max\Bigg(-c_j(\mathbf{x}), 10^{-10}\Bigg)\Bigg)\Bigg)$$

$$Y(\mathbf{x}) = Y_f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\log\Bigg(\max\Bigg(-Y_{c_j}(\mathbf{x}), 10^{-10}\Bigg)\Bigg)$$

Constrained BO: GP-regression based methods

- Barrier method (BM) $\rightarrow$ EI-BM, UCB-BM, Mean-BM:

$$B(\mathbf{x};\gamma) = f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\Bigg(\log\Bigg(\max\Bigg(-c_j(\mathbf{x}), 10^{-10}\Bigg)\Bigg)\Bigg)$$

$$Y(\mathbf{x}) = Y_f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\log\Bigg(\max\Bigg(-Y_{c_j}(\mathbf{x}), 10^{-10}\Bigg)\Bigg)$$

Set $1/\gamma = \sigma_f^2$, and $\mathbb{E}(Y(\mathbf{x})) = m_f(\mathbf{x}) - A$

Constrained BO: GP-regression based methods

- Barrier method (BM) → EI-BM, UCB-BM, Mean-BM:

$$B(\mathbf{x};\gamma) = f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\left(\log\left(\max\left(-c_j(\mathbf{x}), 10^{-10}\right)\right)\right)$$

$$Y(\mathbf{x}) = Y_f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\log\left(\max\left(-Y_{c_j}(\mathbf{x}), 10^{-10}\right)\right)$$

Set $1/\gamma = \sigma_f^2$, and $\mathbb{E}(Y(\mathbf{x})) = m_f(\mathbf{x}) - A$

$$A = \sigma_f^2\sum_{j=1}^{m}\left(\log\left(\max\left(-m_{c_j}(\mathbf{x}), 10^{-10}\right)\right) + \frac{\sigma_{c_j}^2(\mathbf{x})}{2m_{c_j}^2(\mathbf{x})}\right)$$

Constrained BO: GP-regression based methods

- Barrier method (BM) $\rightarrow$ EI-BM, UCB-BM, Mean-BM:

$$B(\mathbf{x};\gamma) = f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\left(\log\left(\max\left(-c_j(\mathbf{x}), 10^{-10}\right)\right)\right)$$

$$Y(\mathbf{x}) = Y_f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\log\left(\max\left(-Y_{c_j}(\mathbf{x}), 10^{-10}\right)\right)$$

Set $1/\gamma = \sigma_f^2$, and $\mathbb{E}(Y(\mathbf{x})) = m_f(\mathbf{x}) - A$

$$A = \sigma_f^2\sum_{j=1}^{m}\left(\log\left(\max\left(-m_{c_j}(\mathbf{x}), 10^{-10}\right)\right) + \frac{\sigma_{c_j}^2(\mathbf{x})}{2m_{c_j}^2(\mathbf{x})}\right)$$

$$\alpha_{\text{Mean-BM}}(\mathbf{x}) = -m_f(\mathbf{x}) + A$$

Constrained BO: GP-regression based methods

- Barrier method (BM) $\rightarrow$ EI-BM, UCB-BM, Mean-BM:

$$B(\mathbf{x};\gamma) = f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\Bigg(\log\Bigg(\max\Bigg(-c_j(\mathbf{x}), 10^{-10}\Bigg)\Bigg)\Bigg)$$

$$Y(\mathbf{x}) = Y_f(\mathbf{x}) - \frac{1}{\gamma}\sum_{j=1}^{m}\log\Bigg(\max\Bigg(-Y_{c_j}(\mathbf{x}), 10^{-10}\Bigg)\Bigg)$$

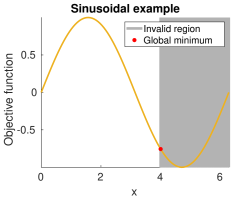
Set $1/\gamma = \sigma_f^2$, and $\mathbb{E}(Y(\mathbf{x})) = m_f(\mathbf{x}) - A$

$$A = \sigma_f^2\sum_{j=1}^{m}\Bigg(\log\Bigg(\max\Bigg(-m_{c_j}(\mathbf{x}), 10^{-10}\Bigg)\Bigg) + \frac{\sigma_{c_j}^2(\mathbf{x})}{2m_{c_j}^2(\mathbf{x})}\Bigg)$$

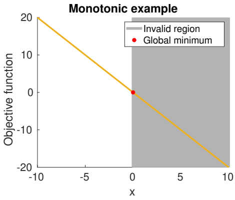
$$\alpha_{\text{Mean-BM}}(\mathbf{x}) = -m_f(\mathbf{x}) + A$$

$$\alpha_{\text{EI-BM/UCB-BM}}(\mathbf{x}) = \alpha_{\text{EI/UCB}}(\mathbf{x}) + A$$

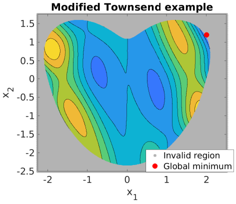
Benchmark examples



(a)

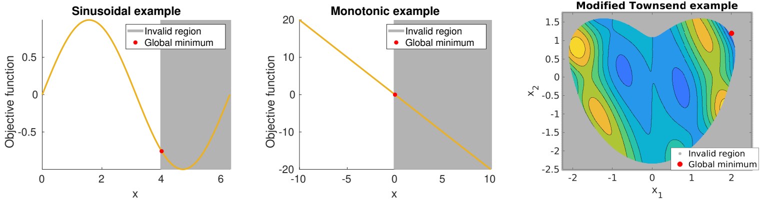


(b)



(c)

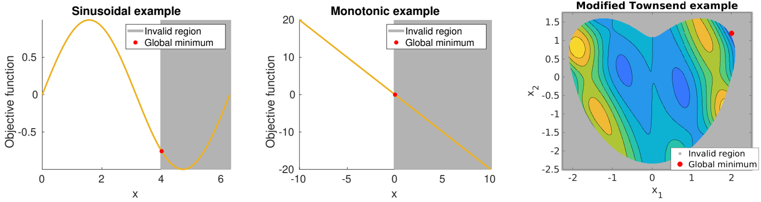
Benchmark examples



(a) (b) (c)

$$\text{(a)}: f(x) = \sin(x), \quad c(x) = x - 4, \quad 0 \leq x \leq 2\pi$$

Benchmark examples

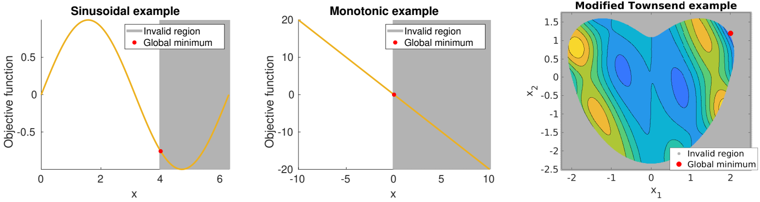


(a) (b) (c)

$$\text{(a)}: f(x) = \sin(x), \quad c(x) = x - 4, \quad 0 \leq x \leq 2\pi$$

$$\text{(b)}: f(x) = -2x, \quad c(x) = x, \quad -10 \leq x \leq 10$$

Benchmark examples



(a) (b) (c)

$$(a): f(x) = \sin(x), \quad c(x) = x - 4, \quad 0 \le x \le 2\pi$$

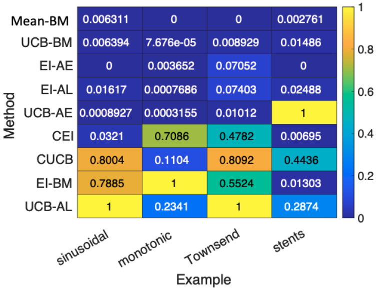
$$(b): f(x) = -2x, \quad c(x) = x, \quad -10 \le x \le 10$$

$$(c): f(x_1, x_2) = -(\cos((x_1 - 0.1)x_2))^2 - x_1 \sin(3x_1 + x_2),$$

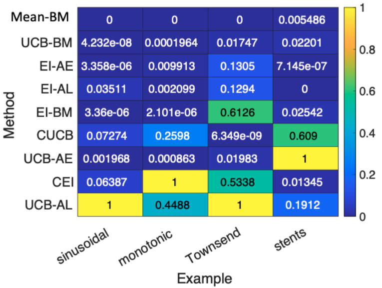
$$c(x_1, x_2) = x_1^2 + x_2^2 - \left(2\cos(t) - \frac{1}{2}\cos(2t) - \frac{1}{4}\cos(3t) - \frac{1}{8}\cos(4t)\right)^2 - (2\sin(t))^2$$

$$t = \arctan\left(\frac{x_1}{x_2}\right), \quad -2.25 \le x_1 \le 2.5, \quad -2.5 \le x_2 \le 1.75$$

Method comparison



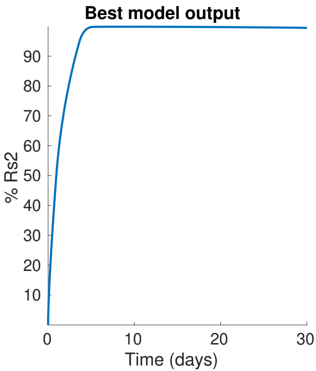
Accuracy

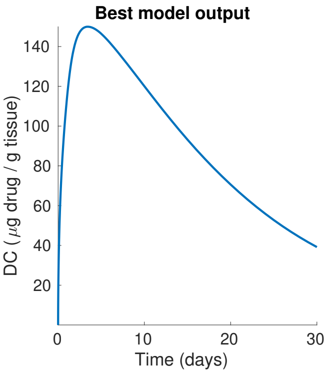


Accuracy-Efficiency

Accuracy: Incumbent minimum objective function (OF) value
Accuracy-efficiency: low OF value and low % of points in the critical region

Stents optimisation results





Final remarks

- We have employed constrained Bayesian optimisation to tackle the high computing times of the stents model and a difficult constrained optimisation problem, with the constrained global optimum at the constraint boundary.

Final remarks

- We have employed constrained Bayesian optimisation to tackle the high computing times of the stents model and a difficult constrained optimisation problem, with the constrained global optimum at the constraint boundary.
- We have performed an assessment of these methods with respect to accuracy and efficiency on several problems.

Final remarks

- We have employed constrained Bayesian optimisation to tackle the high computing times of the stents model and a difficult constrained optimisation problem, with the constrained global optimum at the constraint boundary.
- We have performed an assessment of these methods with respect to accuracy and efficiency on several problems.
- Best average method is Mean-BM wrt both accuracy and accuracy-efficiency

Final remarks

- We have employed constrained Bayesian optimisation to tackle the high computing times of the stents model and a difficult constrained optimisation problem, with the constrained global optimum at the constraint boundary.
- We have performed an assessment of these methods with respect to accuracy and efficiency on several problems.
- Best average method is Mean-BM wrt both accuracy and accuracy-efficiency
- No single best method across all applications