Robust PDE Constrained Optimization with Multilevel Monte Carlo Methods

Andreas Van Barel, Stefan Vandewalle

KU Leuven - University of Leuven, Dept of Computerscience
Section of Numerical and Applied Mathematics
Celestijnenlaan 200A, B3001 Leuven, Belgium

WSC Spring Meeting, June 1st, 2018

Overview

1. Robust control problem
2. Optimality conditions
3. Effect of the variance estimator
4. Multilevel Monte Carlo sampling
5. Optimization with uncertainties
6. Numerical results

Overview

1. Robust control problem
2. Optimality conditions
3. Effect of the variance estimator
4. Multilevel Monte Carlo sampling
5. Optimization with uncertainties
6. Numerical results

Robust control problem

Deterministic problem

$$\min_{y,u} \quad J_{\text{det}}(y,u) \ \ \text{s.t.} \quad c(y,u,k) = 0$$

Robust control problem

$$\min_{y,u} \quad J(y,u) = \mathbb{E}[J_{\text{det}}(y,u)] + \gamma \|\mathbb{S}[y]\|^2 \ \ \text{s.t.} \quad c(y,u,k) = 0$$

$\gamma > 0$ and J_{det} convex $\Rightarrow$ J convex

Note: Other risk measures exist

Common case: tracking target y_D

$$\begin{aligned} J(y,u) &= \mathbb{E}[\|y - y_D\|^2] + \alpha\|u\|^2 + \gamma\|\mathbb{S}[y]\|^2 \\ &= \|\mathbb{E}[y] - y_D\|^2 + \alpha\|u\|^2 + (\gamma+1)\|\mathbb{S}[y]\|^2 \end{aligned}$$

Equivalence of robust and average control for 2-norm tracking

$\gamma > -1 \Rightarrow J$ convex and quadratic

Overview

Optimality conditions

Lagrangian $\mathcal{L}(y,u,p) = J(y,u) + (p, c(y,u,k))$

$$\begin{cases} \nabla_p \mathcal{L} = 0 \\ \nabla_y \mathcal{L} = 0 \\ \nabla_u \mathcal{L} = 0 \end{cases} \Rightarrow \begin{cases} 0 = c(y,u) & \leftarrow \text{constraint equation} \\ 0 = \left(\dfrac{\partial c}{\partial y}\right)^* [p] + \nabla_y J & \leftarrow \text{adjoint equation} \\ 0 = \left(\dfrac{\partial c}{\partial u}\right)^* [p] + \nabla_u J \ = \nabla \tilde{J}(u) \end{cases}$$

$\tilde{J}(u)$ is the reduced cost functional (y eliminated using c)

Optimality conditions

Lagrangian $\mathcal{L}(y,u,p) = J(y,u) + (p, c(y,u,k))$

$$\begin{cases} \nabla_p \mathcal{L} = 0 \\ \nabla_y \mathcal{L} = 0 \\ \nabla_u \mathcal{L} = 0 \end{cases} \Rightarrow \begin{cases} 0 = c(y,u) & \leftarrow \text{constraint equation} \\ 0 = \left(\dfrac{\partial c}{\partial y}\right)^* [p] + \nabla_y J & \leftarrow \text{adjoint equation} \\ 0 = \left(\dfrac{\partial c}{\partial u}\right)^* [p] + \nabla_u J = \nabla \tilde{J}(u) \end{cases}$$

$\tilde{J}(u)$ is the reduced cost functional (y eliminated using c)

If $J(y,u) = \mathbb{E}[J_{\text{det}}(y,u)] + \gamma \|\mathbb{S}[y]\|^2$

$$\begin{cases} c(y,u) = 0 \\ -\left(\dfrac{\partial c}{\partial y}\right)^* [p] = \nabla_y J_{\text{det}} + 2\gamma(y - \mathbb{E}[y]) \\ \nabla \tilde{J}(u) = \left(\dfrac{\partial c}{\partial u}\right)^* [p] + \mathbb{E}[\nabla_u J_{\text{det}}] = 0 \end{cases}$$

$\left(\frac{\partial c}{\partial u}\right)^* [p]$ evaluates to something of the form $\mathbb{E}[f(p)]$

Overview

Estimators for the variance $\mathbb{V}[y]$

Let Ω be the set of all possible random realizations ω (k and y depend on ω)

- Estimator $\hat{V}[y]$ using samples $\Omega_0 \subset \Omega$

$$\hat{V}_0[y] \triangleq \frac{1}{n}\sum_{j=1}^{n}(y_j - \frac{1}{n}\sum_{i=1}^{n} y_i)^2$$

$$\nabla_y \left\| \sqrt{\hat{V}_0[y]} \right\|^2 = 2(y - \frac{1}{n}\sum_{i=1}^{n} y_i)$$

(proof: $\nabla_y \|\mathbb{S}[y]\|^2 = \gamma(y - \mathbb{E}[y])$ holds for any stochastic space, and therefore also for a finite subset of n equally probable samples $\Omega_0 \subset \Omega$ $\square$)

Estimators for the variance $\mathbb{V}[y]$

Let Ω be the set of all possible random realizations ω (k and y depend on ω)

- Estimator $\hat{V}[y]$ using samples $\Omega_0 \subset \Omega$

$$\hat{V}_0[y] \triangleq \frac{1}{n} \sum_{j=1}^{n} (y_j - \frac{1}{n} \sum_{i=1}^{n} y_i)^2$$

$$\nabla_y \left\| \sqrt{\hat{V}_0[y]} \right\|^2 = 2(y - \frac{1}{n} \sum_{i=1}^{n} y_i)$$

(proof: $\nabla_y \|\mathbb{S}[y]\|^2 = \gamma(y - \mathbb{E}[y])$ holds for any stochastic space, and therefore also for a finite subset of n equally probable samples $\Omega_0 \subset \Omega$ $\square$)

Using $\hat{V}[y]$ corresponds to using MC estimator for $\mathbb{E}[y]$

Estimators for the variance $\mathbb{V}[y]$

Let Ω be the set of all possible random realizations ω
(k and y depend on ω)

- Estimator $\hat{V}[y]$ using samples $\Omega_0 \subset \Omega$

$$\hat{V}_0[y] \triangleq \frac{1}{n} \sum_{j=1}^{n} (y_j - \frac{1}{n} \sum_{i=1}^{n} y_i)^2$$

$$\nabla_y \left\| \sqrt{\hat{V}_0[y]} \right\|^2 = 2(y - \frac{1}{n} \sum_{i=1}^{n} y_i)$$

(proof: $\nabla_y \|\mathbb{S}[y]\|^2 = \gamma(y - \mathbb{E}[y])$ holds for any stochastic space, and therefore also for a finite subset of n equally probable samples $\Omega_0 \subset \Omega$ $\square$)

Using $\hat{V}[y]$ corresponds to using MC estimator for $\mathbb{E}[y]$

Problems: - Either large memory required or double work
- Which accuracy to request for $\mathbb{E}[y]$?

Estimators for the variance $\mathbb{V}[y]$

Let Ω be the set of all possible random realizations ω
(k and y depend on ω)

- Another estimator $\hat{V}'[y]$ also using samples $\Omega_0 \subset \Omega$

$$\hat{V}'[y] \triangleq \frac{1}{2n} \sum_{j=1}^{n} (y_j - y_{j-1})^2$$

$$\nabla_y \left\| \sqrt{\hat{V}'[y]} \right\|^2 = 2y - y_{+1} - y_{-1}$$

The j-th sample is $\gamma(2y_j - y_{j+1} - y_{j-1})$ with $y_{n+i} = y_i$

$\hat{V}'[y]$ is an unbiased estimator for $\mathbb{V}[y]$
(proof: $\mathbb{E}[(y_j - y_{j-1})^2] = \mathbb{E}[(y_j - \mathbb{E}[y] + \mathbb{E}[y] - y_{j-1})^2] =$
$\mathbb{E}[(y_j - \mathbb{E}[y])^2] + \mathbb{E}[(\mathbb{E}[y] - y_{j-1})^2] = 2\mathbb{V}[y]$ $\square$)

Estimators for the variance $\mathbb{V}[y]$

Let Ω be the set of all possible random realizations ω (k and y depend on ω)

- Another estimator $\hat{V}'[y]$ also using samples $\Omega_0 \subset \Omega$

$$\hat{V}'[y] \triangleq \frac{1}{2n} \sum_{j=1}^{n} (y_j - y_{j-1})^2$$

$$\nabla_y \left\| \sqrt{\hat{V}'[y]} \right\|^2 = 2y - y_{+1} - y_{-1}$$

The j-th sample is $\gamma(2y_j - y_{j+1} - y_{j-1})$ with $y_{n+i} = y_i$

$\hat{V}'[y]$ is an unbiased estimator for $\mathbb{V}[y]$
(proof: $\mathbb{E}[(y_j - y_{j-1})^2] = \mathbb{E}[(y_j - \mathbb{E}[y] + \mathbb{E}[y] - y_{j-1})^2] = \mathbb{E}[(y_j - \mathbb{E}[y])^2] + \mathbb{E}[(\mathbb{E}[y] - y_{j-1})^2] = 2\mathbb{V}[y] \;\square$)

(+) No more $\mathbb{E}[.]$ required in advance.
(-) Samples are no longer independent!
(-) RMSE($\hat{V}'[y]$) $= 1.5 \cdot$ RMSE($\hat{V}[y]$).

Overview

1. Robust control problem
2. Optimality conditions
3. Effect of the variance estimator
4. Multilevel Monte Carlo sampling
5. Optimization with uncertainties
6. Numerical results

Multilevel Monte Carlo Sampling

Assume a quantity of interest Q (e.g., point value of p)

Multilevel Monte Carlo idea

- Multiple discretization levels $m_0 < m_1 < \ldots < m_L$
- Multiple approximations $Q_{m_0}, \ldots, Q_{m_L}$ for Q
- Telescopic sum

$$\mathbb{E}[Q_{m_L}] = \mathbb{E}[Q_{m_0}] + \sum_{\ell=1}^{L} \mathbb{E}[Q_{m_\ell} - Q_{m_{\ell-1}}] = \sum_{\ell=0}^{L} \mathbb{E}[Y_\ell]$$

Multilevel Monte Carlo estimator

$$\hat{Q}_{\boldsymbol{m},\boldsymbol{n}}^{\mathsf{MLMC}} \triangleq \sum_{\ell=0}^{L} \hat{Y}_{\ell,n_\ell}^{\mathsf{MC}} \quad \text{with} \quad \hat{Y}_{\ell,n_\ell}^{\mathsf{MC}} = \frac{1}{n_\ell} \sum_{i=1}^{n_\ell} \Big(Q_{m_\ell}(\omega_i) - Q_{m_{\ell-1}}(\omega_i) \Big)$$

Multilevel Monte Carlo Sampling

Mean Square Error

$$\mathbb{E}\Big[\big(\hat{Q}^{\text{MLMC}}_{\boldsymbol{m},\boldsymbol{n}} - \mathbb{E}[Q]\big)^2\Big] = \underbrace{\mathbb{V}[\hat{Q}^{\text{MLMC}}_{\boldsymbol{m},\boldsymbol{n}}]}_{\text{stochastic error}} + \underbrace{(\mathbb{E}[Q_{m_L}] - \mathbb{E}[Q])^2}_{\text{bias = discretization error}} \leq \epsilon^2$$

Independent samples

$$\mathbb{V}[\hat{Q}^{\text{MLMC}}_{\boldsymbol{m},\boldsymbol{n}}] = \sum_{\ell=0}^{L} \mathbb{V}[\hat{Y}^{\text{MC}}_{\ell,n_\ell}] = \sum_{\ell=0}^{L} n_\ell^{-1} \mathbb{V}[Y_\ell]$$

- Amount of levels L incremented until bias (estimated) is small enough.
- Amount of samples $\boldsymbol{n} = (n_0, n_1, \ldots, n_L)$ chosen such that stochastic error is small enough.

Multilevel Monte Carlo Sampling

Mean Square Error

$$\mathbb{E}\left[\left(\hat{Q}^{\mathsf{MLMC}}_{\boldsymbol{m},\boldsymbol{n}} - \mathbb{E}[Q]\right)^2\right] = \underbrace{\mathbb{V}[\hat{Q}^{\mathsf{MLMC}}_{\boldsymbol{m},\boldsymbol{n}}]}_{\text{stochastic error}} + \underbrace{(\mathbb{E}[Q_{m_L}] - \mathbb{E}[Q])^2}_{\text{bias} \,=\, \text{discretization error}} \leq \epsilon^2$$

Dependent samples

$$\mathbb{V}[\hat{Q}^{\mathsf{MLMC}}_{\boldsymbol{m},\boldsymbol{n}}] = \sum_{\ell=0}^{L} \mathbb{V}[\hat{Y}^{\mathsf{MC}}_{\ell,n_\ell}] = \sum_{\ell=0}^{L} n_\ell^{-2} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_\ell} \mathsf{Cov}[Y_{\ell,i}, Y_{\ell,j}]$$

- Amount of levels L incremented until bias (estimated) is small enough.
- Amount of samples $\boldsymbol{n} = (n_0, n_1, \ldots, n_L)$ chosen such that stochastic error is small enough.

Multilevel Monte Carlo Sampling

Mean Square Error

$$\mathbb{E}\left[\left(\hat{Q}^{\mathsf{MLMC}}_{\boldsymbol{m},\boldsymbol{n}} - \mathbb{E}[Q]\right)^2\right] = \underbrace{\mathbb{V}[\hat{Q}^{\mathsf{MLMC}}_{\boldsymbol{m},\boldsymbol{n}}]}_{\text{stochastic error}} + \underbrace{(\mathbb{E}[Q_{m_L}] - \mathbb{E}[Q])^2}_{\text{bias = discretization error}} \leq \epsilon^2$$

Dependent samples (circulant covariance matrix)

$$\mathbb{V}[\hat{Q}^{\mathsf{MLMC}}_{\boldsymbol{m},\boldsymbol{n}}] = \sum_{\ell=0}^{L} \mathbb{V}[\hat{Y}^{\mathsf{MC}}_{\ell,n_\ell}] = \sum_{\ell=0}^{L} n_\ell^{-1}(\mathbb{V}[Y_\ell] + 2\sum_{j=2}^{b+1} \mathsf{Cov}[Y_{\ell,1}, Y_{\ell,j}])$$

- Amount of levels L incremented until bias (estimated) is small enough.
- Amount of samples $\boldsymbol{n} = (n_0, n_1, \ldots, n_L)$ chosen such that stochastic error is small enough.

Multilevel Monte Carlo Sampling

Theorem (MLMC cost[1])

Assumptions (slightly simplified)

$$|\mathbb{E}[Q_{m_\ell} - Q]| \lesssim m_\ell^{-\rho} \qquad \mathbb{V}[Y_\ell] \lesssim m_\ell^{-\phi} \qquad \mathcal{C}_\ell \lesssim m_\ell^{\kappa}$$

MLMC cost

$$\mathcal{C}_{MLMC}(\epsilon) \lesssim \begin{cases} \epsilon^{-2} & \textit{if } \phi > \kappa \\ \epsilon^{-2}(\log \epsilon)^2 & \textit{if } \phi = \kappa \\ \epsilon^{-2-(\kappa-\phi)/\rho} & \textit{if } \phi < \kappa \end{cases}$$

For dependent samples: replace $\mathbb{V}[Y_\ell]$ as suggested by the previous slides. Note: in the last case, no amendmend is needed since

$$\mathbb{V}[Y_\ell] \lesssim m_\ell^{-\phi} \Rightarrow \mathbb{V}[Y_\ell] + 2\sum_{j=2}^{b+1} \mathrm{Cov}[Y_{\ell,1}, Y_{\ell,j}] \lesssim m_\ell^{-\phi}$$

[1] K. A. Cliffe, M. B. Giles, R. Scheichl, *Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients*, Computing and Visualization in Science, vol. 14(1), pp. 3–15, 2011.

Model problem

Classical model problem

- Diffusion PDE constraint $c(y, u, k) = 0$

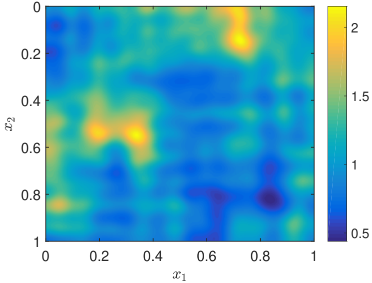
$$\begin{aligned} -\nabla \cdot (k(x,\omega)\nabla y(x,\omega)) &= u(x) \quad \text{on } D \\ y(x,\omega) &= 0 \qquad\quad \text{on } \partial D \end{aligned}$$

- lognormal random field $k(x,\omega) = \exp(z(x,\omega))$ with z Gaussian. $\mathbb{E}[z(x,\omega)] = 0$ and, e.g.,

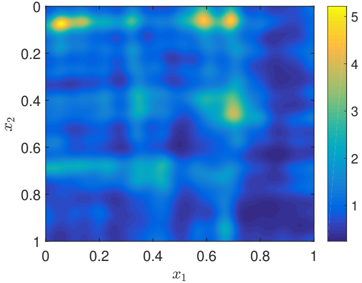
$$\mathrm{Cov}(z(x_1,\omega), z(x_2,\omega)) = \sigma^2 \exp\left(-\frac{\|x_1 - x_2\|_1}{\lambda}\right)$$

- Robust tracking type cost: $\mathbb{E}[\|y - y_D\|^2] + \alpha\|u\|^2 + \gamma\|\mathbb{S}[y]\|^2$

Samples of k



(a) $\sigma^2 = 0.1,\ \lambda = 0.3$



(b) $\sigma^2 = 0.5,\ \lambda = 0.3$

generated using, e.g., the KL-expansion or circulant embedding

Optimality conditions, gradient and Hessian

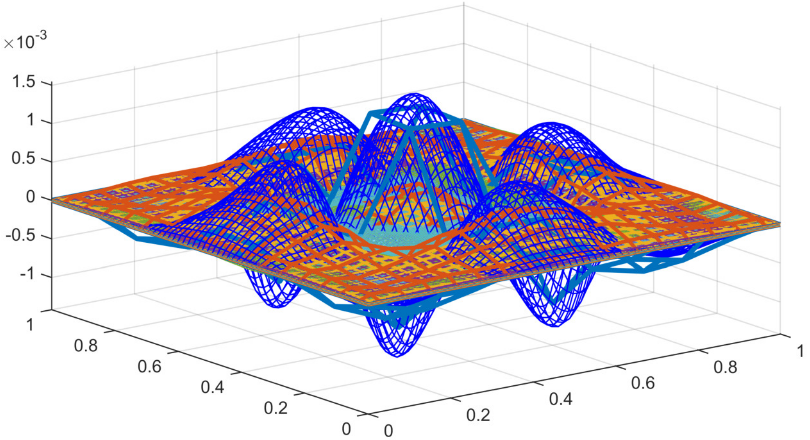
Gradient: $\nabla \tilde{J}(u)$

$$\left\{ \begin{array}{rcl} -\nabla \cdot (k\nabla y) & = & u \\ -\nabla \cdot (k\nabla p) & = & 2(y - y_D) + 2\gamma(y - \mathbb{E}[y]) \\ \nabla \tilde{J}(u) & = & 2\alpha u + \beta\mathbb{E}[p] \end{array} \right.$$

Hessian: Hess $\tilde{J}(u)[\delta u]$

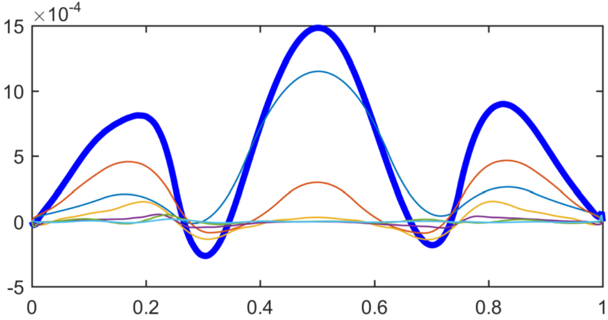
$$\left\{ \begin{array}{rcl} -\nabla \cdot (k\nabla\, \delta y) & = & \delta u \\ -\nabla \cdot (k\nabla\, \delta p) & = & 2\,\delta y + 2\gamma(\delta y - \mathbb{E}[\delta y]) \\ \text{Hess }\tilde{J}(u)[\delta u] & = & 2\alpha\,\delta u + \mathbb{E}[\delta p] \end{array} \right.$$

Multilevel decomposition of gradient



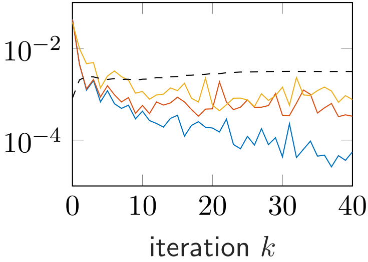
Multilevel decomposition of gradient

Cross section of gradient contributions after mapping to the finest level.

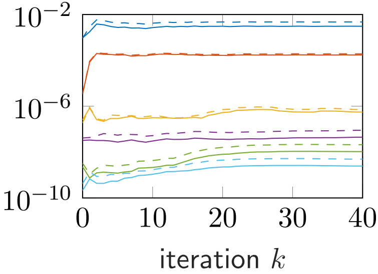


Overview

Behavior of optimization and evolution of variances



(a) $\|\nabla \hat{J}_f(\boldsymbol{u}_f^{(k)})\|$ (——),
$\|\nabla \hat{J}_\$(\boldsymbol{u}_f^{(k)})\|$ (——),
$\|\nabla \hat{J}_\$(\boldsymbol{u}_\$^{(k)})\|$ (——),
RMSE ϵ of $\nabla \hat{J}_f(\boldsymbol{u}_f^{(k)})$ (- - -).



(b) $\|\mathbb{V}[\boldsymbol{Y}_\ell]\|_\infty$ (- - -),
$\|\max\{\frac{1}{2}\mathbb{V}[\boldsymbol{Y}_\ell],$
$\mathbb{V}[\boldsymbol{Y}_\ell] + 2\sum_{j=2}^{b+1} \mathsf{Cov}[\boldsymbol{Y}_{\ell,1}, \boldsymbol{Y}_{\ell,j}]\}\|_\infty$
(——)
for levels $\ell = \{0, \ldots, 5\}$.

Convergence behaviour

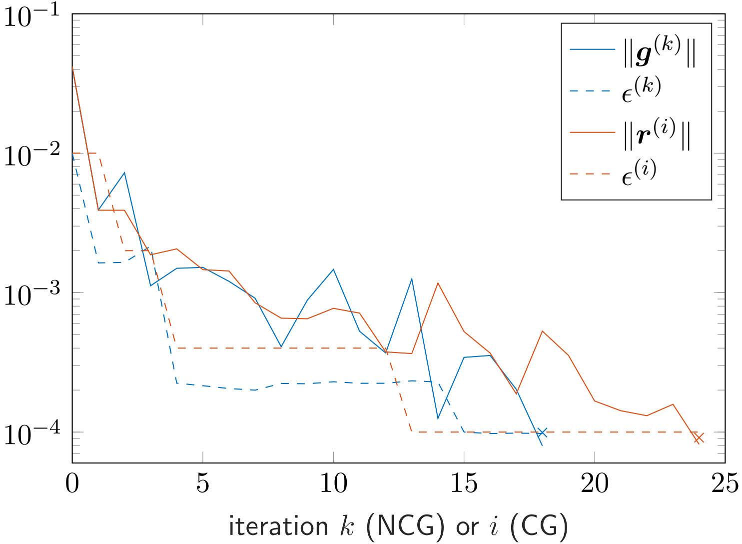


Figure: Behavior of gradient (NCG) and Hessian (CG) based optimization

MLMC optimization cost

Theorem (MLMC optimization cost)

Assumptions Same assumptions but uniformly in an area of the optimal point $\bar{u}$ and uniformly for each point of the gradient:

$$|\mathbb{E}[Q_{m_\ell} - Q]| \lesssim m_\ell^{-\rho} \qquad \mathbb{V}[Y_\ell] \lesssim m_\ell^{-\phi} \qquad \mathcal{C}_\ell \lesssim m_\ell^{\kappa}$$

Optimization cost (using the gradient or Hessian based algorithm) to reach gradient norm τ

$$\mathcal{C}_{opt}(\tau) \lesssim \begin{cases} \tau^{-2} & \text{if } \phi > \kappa \\ \tau^{-2}(\log \tau)^2 & \text{if } \phi = \kappa \\ \tau^{-2-(\kappa-\phi)/\rho} & \text{if } \phi < \kappa \end{cases}, \quad \tau \to 0. \tag{1}$$

Again, for dependent samples, replace $\mathbb{V}[Y_\ell]$ as suggested by the previous slides.

Overview

Example

problem description		solver parameters	
$\alpha = 1\mathrm{e}{-6}$	$\sigma^2 = 0.1$	$m_0 = 8$	$\bar{L} = 5$
$\gamma = 1$	$\lambda = 0.3$	$m_{\bar{L}} = 256$	$\mathsf{TOL} = 1\mathrm{e}{-4}$

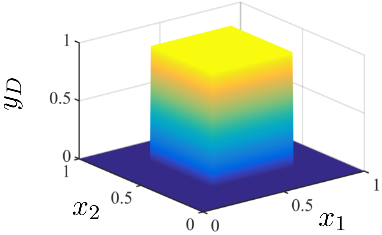


Figure: Target function y_D.

Example

Gradient based algorithm:

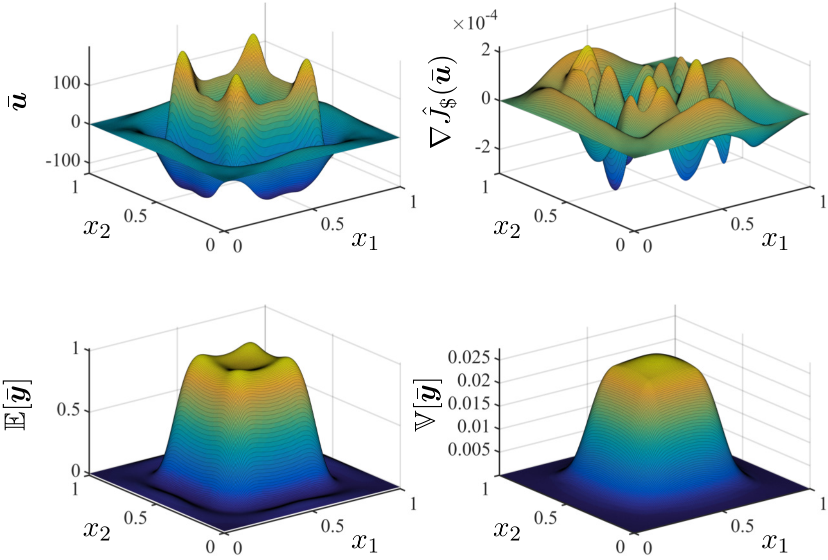
Total time: 1542s

k	$\epsilon^{(k)}$	n_0	n_1	n_2	n_3	n_4	n_5	estimate of ρ	$t^{(k)}$
0	0.01	140	76	44				2.0237	2.05
4	2.24e−4	17150	1512	80	28	20		1.5824	47.49
15	1e−4	98452	9156	940	118	20		1.5825	248.84

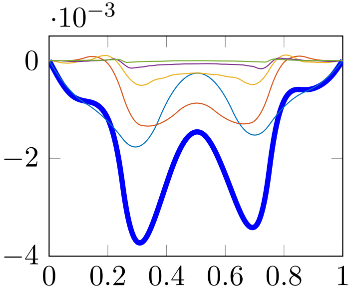
Hessian based algorithm:

Total time: 1989s

i	$\epsilon^{(i)}$	n_0	n_1	n_2	n_3	n_4	n_5	estimate of ρ	$t^{(k)}$
0	0.01	140	76	44				1.8355	8.25
2	2e−3	140	76	44				1.703	8.73
4	4e−4	5964	521	44	28	20		1.5905	130.11
13	1e−4	96159	9010	821	93	22		1.6195	1841.44


$\bar{\boldsymbol{u}}$
100
0
-100
1
0.5
0
x_2
0
0.5
1
x_1
$\times 10^{-4}$
2
0
-2
$\nabla \hat{J}_{\$}(\bar{\boldsymbol{u}})$
$\mathbb{E}[\bar{\boldsymbol{y}}]$
1
0.5
0
$\mathbb{V}[\bar{\boldsymbol{y}}]$
0.025
0.02
0.015
0.01
0.005

Example



(a) Iteration $k = 5$ (b) Iteration $k = 18$ (last)

Figure: Cross section of $\boldsymbol{g}^{(k)} = \sum_{\ell=0}^{L} I_{\ell}^{\bar{L}} \hat{Y}_{\ell,n_\ell}^{\mathsf{MC}}$ (——) and contributions $I_{\ell}^{\bar{L}} \hat{Y}_{\ell,n_\ell}^{\mathsf{MC}}$ for levels $0, \ldots, L$ (——, ——, ——, ——, ——).

Nonlinear constraint equation example

- Nonlinear reaction-diffusion problem

$$
\begin{aligned}
-\nabla\cdot(k\nabla y)+f(y) &= u \quad \text{on } D\\
y &= 0 \quad \text{on } \partial D
\end{aligned}
$$

- Gradient (Hessian also possible)

$$
\begin{cases}
-\nabla\cdot(k\nabla y)+f(y) = u & \text{on } D\\
-\nabla\cdot(k\nabla p)+f'(y)p = (1+\gamma)y-y_D-\gamma\mathbb{E}[y] & \text{on } D\\
\nabla\tilde{J}(u) = 2(\alpha u+\beta\mathbb{E}[p]) = 0
\end{cases}
$$

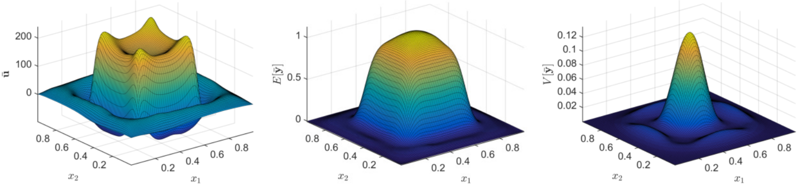
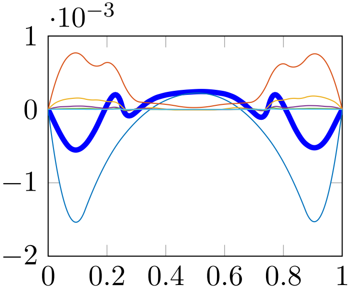
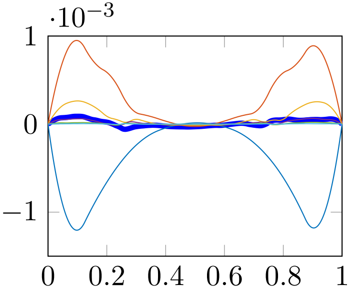


Figure: $\bar{\boldsymbol{u}}$, $\mathbb{E}[\bar{\boldsymbol{y}}]$ and $\mathbb{V}[\bar{\boldsymbol{y}}]$ for $f=e^{5y}+20$. $\sigma^2=0.5$ and $n_{KL}=250$. $\alpha=10^{-7}$, $\gamma=1$. 256×256 finest grid. $\tau=5\text{e}{-5}$.

Nonlinear constraint equation example



(a) Iteration $k = 7$



(b) Iteration $k = 12$ (last)

Figure: Cross section of $\boldsymbol{g}^{(k)} = \sum_{\ell=0}^{L} I_\ell^{\bar{L}} \hat{Y}_{\ell,n_\ell}^{\mathsf{MC}}$ (——) and of contributions $I_\ell^{\bar{L}} \hat{Y}_{\ell,n_\ell}^{\mathsf{MC}}$ at levels $0, \ldots, L$ (——, ——, ——, ——, ——, ——).

Nonlinear constraint equation example

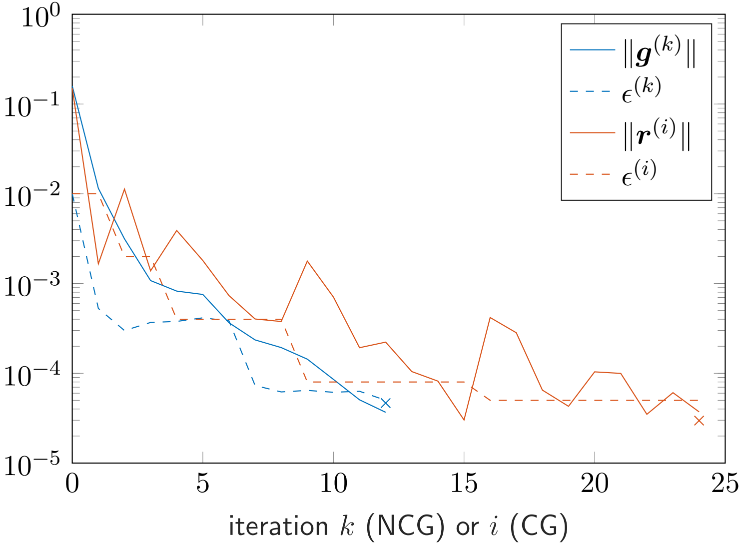


Figure: Behavior of gradient (NCG) and Hessian (CG) based optimization

References

- A. Van Barel, S. Vandewalle, *Robust Optimization of PDEs with Random Coefficients Using a Multilevel Monte Carlo method*, ArXiv preprint 1711.02574, 2017.
- A. A. Ali, E. Ullmann, M. Hinze, *Multilevel Monte Carlo analysis for optimal control of elliptic PDEs with random coefficients*, SIAM/ASA Journal on Uncertainty Quantification, 5 pp. 466–492, 2017.
- D. P. Kouri, *A multilevel stochastic collocation algorithm for optimization of pdes with uncertain coefficients*, SIAM/ASA Journal Uncertainty Quantification, 2, pp. 55–81, 2014.
- K. A. Cliffe, M. B. Giles, R. Scheichl, *Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients*, Computing and Visualization in Science, vol. 14(1), pp. 3–15, 2011.

Questions?

Example 2

problem description		solver parameters	
$\alpha = 1\text{e}{-5}$	$\sigma^2 = 0.5$	$m_0 = 8$	$\bar{L} = 5$
$\gamma = 0$	$\lambda = 0.3$	$m_{\bar{L}} = 256$	$\text{TOL} = 1\text{e}{-4}$

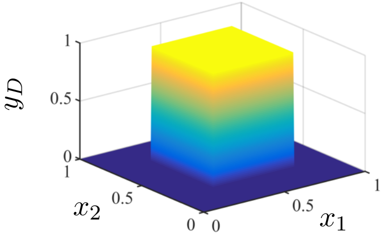


Figure: Target function y_D.

Example 2

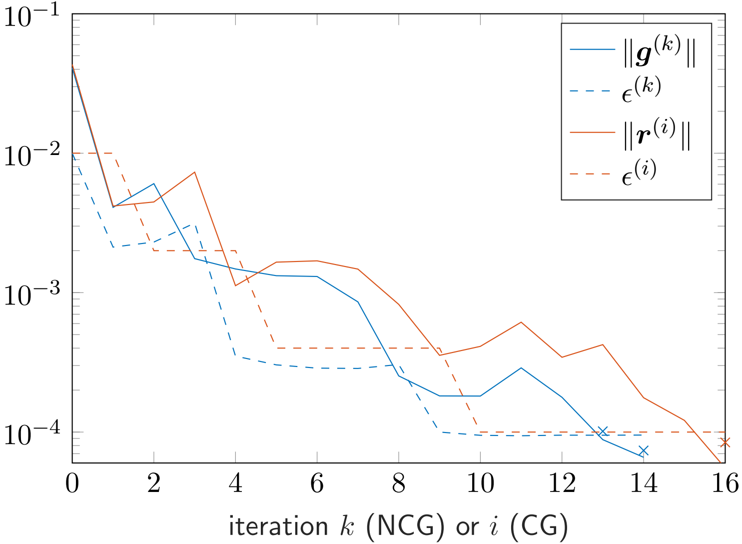


Figure: Behavior of gradient (NCG) and Hessian (CG) based optimization

Example 2

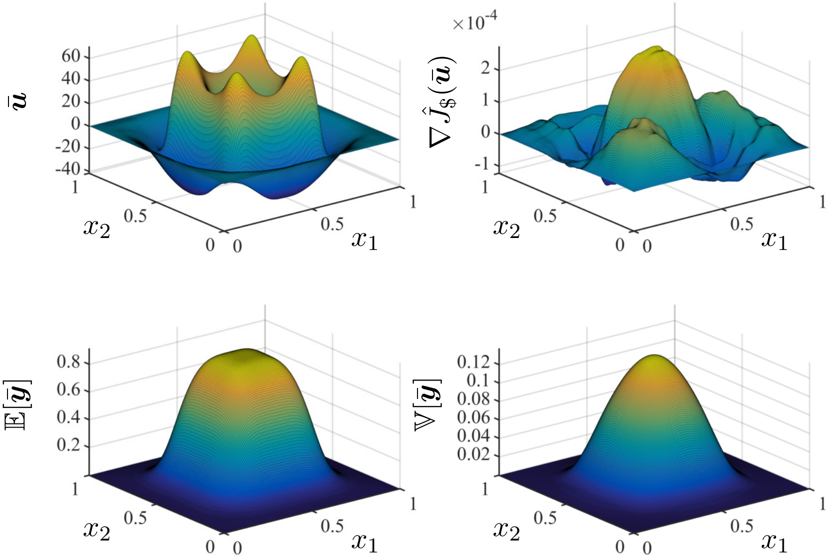
Gradient based algorithm: Total time: 6973s

k	$\epsilon^{(k)}$	n_0	n_1	n_2	n_3	n_4	n_5	estimate of ρ	$t^{(k)}[s]$
0	0.01	140	76	44				2.0237	2.06
4	3.51e−4	35563	3220	136	28	20		1.5824	93.44
9	1e−4	375256	38259	2082	135	21	16	1.5825	1092.71

Hessian based algorithm: Total time: 5114s

i	$\epsilon^{(i)}$	n_0	n_1	n_2	n_3	n_4	n_5	estimate of ρ	$t^{(i)}[s]$
0	0.01	140	76	44				1.9102	8.89
2	2e−3	393	76	44				2.0975	11.77
5	4e−4	33063	6980	193	28	20		1.7029	76.20
10	1e−4	388834	37023	1747	255	56	16	1.7818	668.58

Example 2



$\bar{\boldsymbol{u}}$
60
40
20
0
-20
-40
x_2
x_1
$\times 10^{-4}$
$\nabla \hat{J}_{\$}(\bar{\boldsymbol{u}})$
2
1
0
-1
$\mathbb{E}[\bar{\boldsymbol{y}}]$
0.8
0.6
0.4
0.2
$\mathbb{V}[\bar{\boldsymbol{y}}]$
0.12
0.1
0.08
0.06
0.04
0.02
0
0.5
1