Construction of Gaussian Surrogate Process Using Numerical and Modeling Error Uncertainty

H. Aghakhani, A. K. Patra and E. Spiller

SIAM CSE 2015

March 23, 2015

Table of contents

Complex System Modeling and UQ

- Complex model UQ based on outputs from ensemble of evaluations (costly simulations) used to fit appropriate surrogate – GaSP, Polynomial Chaos ...
- Effect of numerical error in model evaluation on surrogate construction is unclear but examples show **significant and unpredictable** dependence.
- Numerical error is parameter and output feature dependent!

Proposed Approach

Use *a posteriori* error estimates

The dual weighted *a posteriori* numerical approximation error estimate is used to inform surrogate construction.

Complex System Modeling and UQ

- Complex model UQ based on outputs from ensemble of evaluations (costly simulations) used to fit appropriate surrogate – GaSP, Polynomial Chaos ...
- Effect of numerical error in model evaluation on surrogate construction is unclear but examples show **significant and unpredictable** dependence.
- Numerical error is parameter and output feature dependent!

Proposed Approach

Use *a posteriori* error estimates

The dual weighted *a posteriori* numerical approximation error estimate is used to inform surrogate construction.

Numerical Error Estimation

Let:

- $R(U, \alpha) = 0$ is a system of governing equations,
- U is the solution vector, α is the vector of design parameters.

Dual weighted *a posteriori* numerical approximation error estimate needs[1],[2].

- Computation of an approximate residual.

$$R(U, \alpha) - R(U^h, \alpha) = r(U^h, \alpha) \approx r(U^h_H, \alpha)$$

- Computation of an appropriate adjoint.

Complicated for non-linear hyperbolic time dependent systems.

---

[1] Becker et al., An optimal control approach to a posteriori error estimation in finite element methods, Acta Numerica, Jan 2003

[2] Nemec et al., djoint-based adaptive mesh refinement for complex geometries, AIAA Paper, 2008

Discrete Adjoint

- The objective is to minimize error in computing functional $\mathrm{J}(U, \alpha) \quad$ subject to $\quad R(U, \alpha) = 0$
- We can only compute $J(U^h, \alpha)$ from a solution of $R(U^h, \alpha)$
- $r(U^h, \alpha) \rightarrow J(U, \alpha) - J(U^h, \alpha)$
- Adjoint ψ relates $r(U^h, \alpha)$ to $\epsilon(U, U^h, \alpha) \equiv J(U, \alpha) - J(U^h, \alpha)$

Discrete Adjoint Cont.

With writing the variation of the functional and governing equation w.r.t design parameters, we will have:

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial U}\frac{dU}{d\alpha} + \frac{\partial J}{\partial \alpha}$$

and:

Discrete Adjoint Cont.

With writing the variation of the functional and governing equation w.r.t design parameters, we will have:

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial U}\frac{dU}{d\alpha} + \frac{\partial J}{\partial \alpha}$$

and:

$$\frac{\partial R}{\partial U}\frac{dU}{d\alpha} + \frac{\partial R}{\partial \alpha} = 0.$$

Discrete Adjoint Cont.

With writing the variation of the functional and governing equation w.r.t design parameters, we will have:

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial U}\frac{dU}{d\alpha} + \frac{\partial J}{\partial \alpha}$$

and:

$$\frac{\partial R}{\partial U}\frac{dU}{d\alpha} + \frac{\partial R}{\partial \alpha} = 0.$$

Now,if we replace $\frac{dU}{d\alpha}$ from the second equation into the first equation, we have:

$$\frac{dJ}{d\alpha} = -\frac{\partial J}{\partial U}(\frac{\partial R}{\partial U})^{-1}\frac{\partial R}{\partial \alpha} + \frac{\partial J}{\partial \alpha}$$

Discrete Adjoint Cont.

With writing the variation of the functional and governing equation w.r.t design parameters, we will have:

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial U}\frac{dU}{d\alpha} + \frac{\partial J}{\partial \alpha}$$

and:

$$\frac{\partial R}{\partial U}\frac{dU}{d\alpha} + \frac{\partial R}{\partial \alpha} = 0.$$

Now,if we replace $\frac{dU}{d\alpha}$ from the second equation into the first equation, we have:

$$\frac{dJ}{d\alpha} = -\frac{\partial J}{\partial U}(\frac{\partial R}{\partial U})^{-1}\frac{\partial R}{\partial \alpha} + \frac{\partial J}{\partial \alpha}$$

Sensitivity Computation

Assume that n is the size of vector U, and m is the size of vector α:

$$dJ_{scalar} = \underbrace{-\left[\frac{\partial J}{\partial U}\right]_{1\times n}\left[\frac{\partial R}{\partial U}^{-1}\right]_{n\times n}\left[\frac{\partial R}{\partial \alpha}\right]_{n\times m} d\alpha_{m\times 1}}_{\text{scalar}} + \underbrace{\left[\frac{\partial J}{\partial \alpha}\right]_{1\times m} d\alpha_{m\times 1}}_{\text{scalar}}$$

Sensitivity Computation

Assume that n is the size of vector U, and m is the size of vector α:

$$dJ_{scalar} = \underbrace{-\left[\frac{\partial J}{\partial U}\right]_{1\times n}\left[\frac{\partial R}{\partial U}^{-1}\right]_{n\times n}\left[\frac{\partial R}{\partial \alpha}\right]_{n\times m} d\alpha_{m\times 1}}_{\text{scalar}} + \underbrace{\left[\frac{\partial J}{\partial \alpha}\right]_{1\times m} d\alpha_{m\times 1}}_{\text{scalar}}$$

Above sensitivity can be computed in two ways:

1. Forward mode: first computes $(\frac{\partial R}{\partial U})^{-1}\frac{\partial R}{\partial \alpha}$

Sensitivity Computation

Assume that n is the size of vector U, and m is the size of vector α:

$$dJ_{scalar} = \underbrace{-\left[\frac{\partial J}{\partial U}\right]_{1\times n}\left[\frac{\partial R}{\partial U}^{-1}\right]_{n\times n}\left[\frac{\partial R}{\partial \alpha}\right]_{n\times m} d\alpha_{m\times 1}}_{\text{scalar}} + \underbrace{\left[\frac{\partial J}{\partial \alpha}\right]_{1\times m} d\alpha_{m\times 1}}_{\text{scalar}}$$

Above sensitivity can be computed in two ways:

1. Forward mode: first computes $(\frac{\partial R}{\partial U})^{-1}\frac{\partial R}{\partial \alpha}$
2. Adjoint mode: first computes $\frac{\partial J}{\partial U}(\frac{\partial R}{\partial U})^{-1} \rightarrow (\frac{\partial R}{\partial U})^T v = \frac{\partial J}{\partial U}^T$, v is adjoint solution

Sensitivity Computation

Assume that n is the size of vector U, and m is the size of vector α:

$$dJ_{scalar} = \underbrace{-\left[\frac{\partial J}{\partial U}\right]_{1\times n}\left[\frac{\partial R}{\partial U}^{-1}\right]_{n\times n}\left[\frac{\partial R}{\partial \alpha}\right]_{n\times m} d\alpha_{m\times 1}}_{\text{scalar}} + \underbrace{\left[\frac{\partial J}{\partial \alpha}\right]_{1\times m} d\alpha_{m\times 1}}_{\text{scalar}}$$

Above sensitivity can be computed in two ways:

1. Forward mode: first computes $(\frac{\partial R}{\partial U})^{-1}\frac{\partial R}{\partial \alpha}$
2. Adjoint mode: first computes $\frac{\partial J}{\partial U}(\frac{\partial R}{\partial U})^{-1} \rightarrow (\frac{\partial R}{\partial U})^T v = \frac{\partial J}{\partial U}^T$, v is adjoint solution

Advantage of Adjoint:

If the number of design parameters are more than the objective functionals, then the computational cost of the adjoint is much lower than the forward method.

Sensitivity Computation

Assume that n is the size of vector U, and m is the size of vector α:

$$dJ_{scalar} = \underbrace{-\left[\frac{\partial J}{\partial U}\right]_{1\times n}\left[\frac{\partial R}{\partial U}^{-1}\right]_{n\times n}\left[\frac{\partial R}{\partial \alpha}\right]_{n\times m} d\alpha_{m\times 1}}_{\text{scalar}} + \underbrace{\left[\frac{\partial J}{\partial \alpha}\right]_{1\times m} d\alpha_{m\times 1}}_{\text{scalar}}$$

Above sensitivity can be computed in two ways:

1. Forward mode: first computes $(\frac{\partial R}{\partial U})^{-1}\frac{\partial R}{\partial \alpha}$
2. Adjoint mode: first computes $\frac{\partial J}{\partial U}(\frac{\partial R}{\partial U})^{-1} \rightarrow (\frac{\partial R}{\partial U})^T v = \frac{\partial J}{\partial U}^T$, v is adjoint solution

Advantage of Adjoint:

If the number of design parameters are more than the objective functionals, then the computational cost of the adjoint is much lower than the forward method.

Computing Adjoint

To solve adjoint equation we need Transpose of Jacobian Matrix:

$$\boxed{\left(\frac{\partial R}{\partial U}\right)^T v = \frac{\partial J}{\partial U}^T}$$

TITAN2D uses Godunov finite volume with Euler explicit time scheme, so the descritized form of equations are:

$$U_i^{n+1} = U_i^n - \frac{\triangle t}{\triangle x}\{F^n_{i+\frac{1}{2}} - F^n_{i-\frac{1}{2}}\} - \frac{\triangle t}{\triangle y}\{G^n_{i+\frac{1}{2}} - G^n_{i-\frac{1}{2}}\}$$

$$\left(\frac{\partial R}{\partial U}\right)^T_{m\times m} = K_{ij}$$

where m is the number of time steps, and each K_{ij} is a

Computing Adjoint Cont.

For Euler explicit:

$$K_{i,i} = I \quad \text{and} \quad K_{i,i+1} = \left(\frac{\partial R_p^{i+1}}{\partial U_q^i}\right)^T,$$

- p and q are degrees of freedom
- rest of the components are zero
- depend on the stencil is used. K matrices are also block bounded

$$\left(\frac{\partial R}{\partial U}\right)^T_{m\times m} = \begin{pmatrix} I & K_{1,2} & & & \\ & I & K_{2,3} & \mathbf{0} & \\ & & \ddots & \ddots & \\ & \mathbf{0} & & I & K_{m-1,m} \\ & & & & I \end{pmatrix}$$

Computing Adjoint Cont.

Important conclusion:

To compute adjoint for an explicit system, there is no need to solve a system of equation, and adjoint solution can be found marching backward in time.

$$v_1 + K_{1,2} v_2 = \left(\frac{\partial J}{\partial U}\right)_1^T$$

$$\vdots$$

$$v_{m-1} + K_{m-1,m} v_m = \left(\frac{\partial J}{\partial U}\right)_{m-1}^T$$

$$v_m = \left(\frac{\partial J}{\partial U}\right)_m^T$$

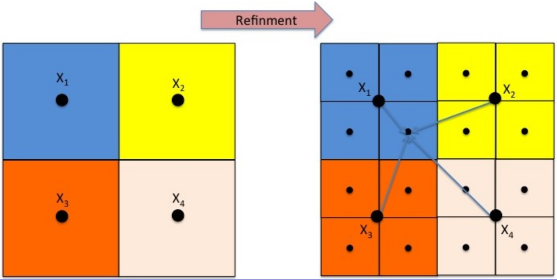
Residual Computation

Recall:

$$R(U,\alpha) - R(U^h,\alpha) = r(U^h,\alpha) \approx r(U_H^h,\alpha)$$

- To compute residual $r(U^h,\alpha)$, we need $R(U^h,\alpha)$ which is expensive.
- But we can find a good approximation for $R(U^h,\alpha)$ from $R(U_H,\alpha)$ which is represented here by $R(U_H^h,\alpha)$.

We approximated $R(U^h,\alpha)$ with bilinear interpolation.



Error Estimation

Let us estimate the error in the objective functional $J(U)$, given the solution on a coarse mesh $J(U_H)$ and our approximate solution $J(U^h, \alpha)$
With Taylor expansion we can write: [3]

$$J(U^h) \approx J(U_H^h) \;-\; \underbrace{(\psi_H^h)^T r(U_H^h)}_{\text{Adjoint correction term}} \;-\; \underbrace{(\psi_h - \psi_H^h)^T r(U_H^h)}_{\text{Remaining term}},$$

where $J(U^h)$ is the functional value on a finer mesh, and all $\Box_H^h$ is the projection of $\Box$ from the coarse mesh to the fine mesh.

[3] Nemec et al., djoint-based adaptive mesh refinement for complex geometries, AIAA Paper, 2008

Error Estimation Cont.

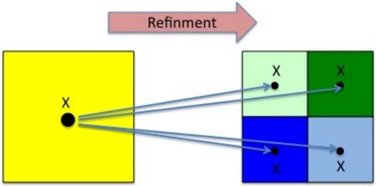
So to compute the error ε

- We have to first find .
- Like $R(U^h, \alpha)$ we can approximate ψ^h with higher order construction.

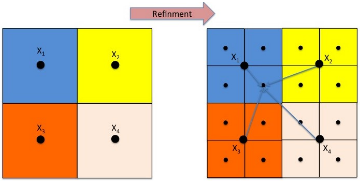
$$J(U^h) \approx J(U_L) - (\psi_L)^T r(U_L) - (\psi_L - \psi_C)^T r(U_L),$$

where $\square_L$, $\square_C$ represent linear and constant reconstruction respectively.

Constant reconstruction



Linear reconstruction



Case 1: Burger's Equation

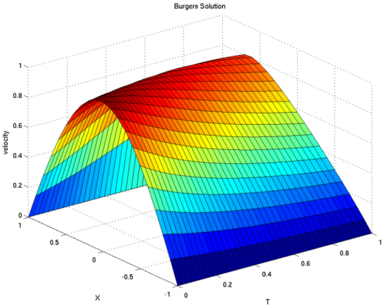
$$
\begin{aligned}
R(x,t) &= \frac{\partial u}{\partial t} + \frac{\alpha}{2}\frac{\partial u^2}{\partial x} = 0, \qquad x \in (-1,1), \quad t \in (0,1),\\
u(x,0) &= \beta \ cos(\frac{\pi}{2}x),\\
u(1,t) &= 0,\\
J &= 0.5 \int_T \int_x u^2 \ dx \ dt,
\end{aligned}
$$

where α and β are uncertain parameters, and are selected from $\mathscr{N}(1, 0.1)$.

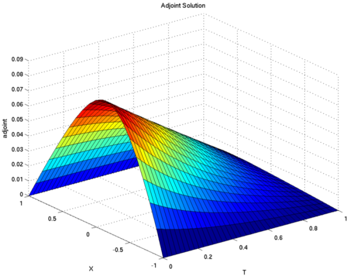
Case 1: Burger's Equation

Results are for a Monte Carlo simulation with 10,000 samples:

Velocity

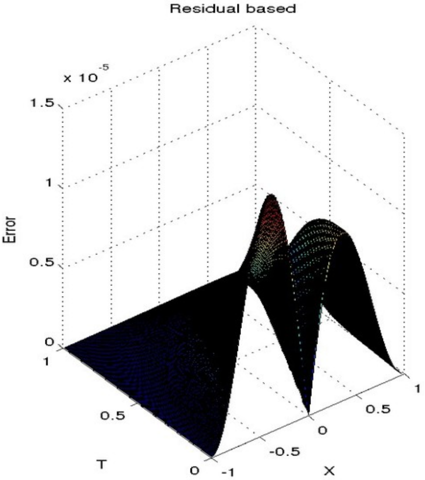


Adjoint



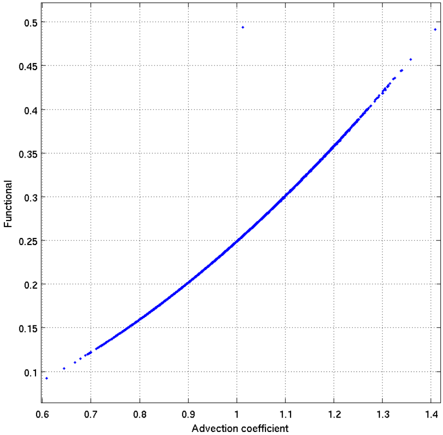
Case 1: Burger's Equation

Computed error



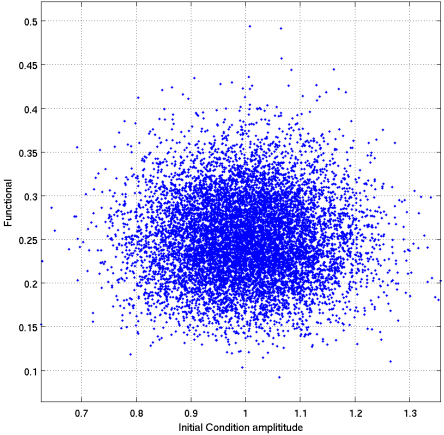
Case 1: Burger's Equation

Functional of interest as a function of advection coefficient



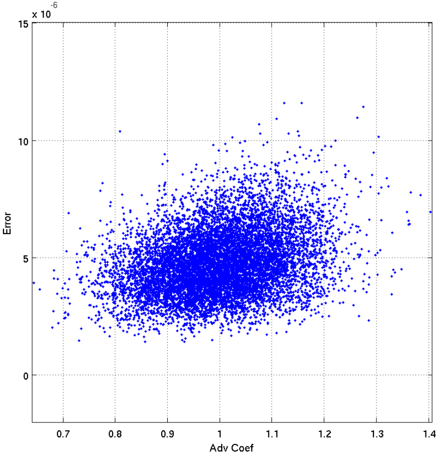
Case 1: Burger's Equation

Functional of interest as a function of initial condition uncertain coefficient



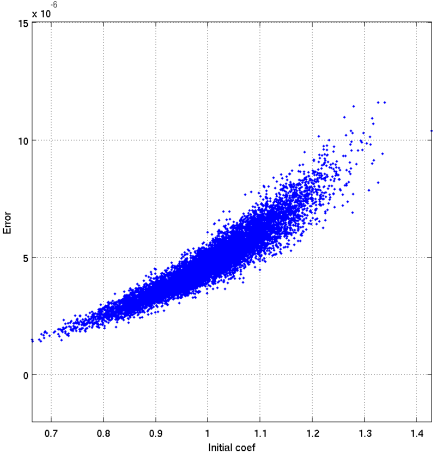
Case 1: Burger's Equation

Error at final stage as a function of advection coefficient



Case 1: Burger's Equation

Error at final stage as a function of initial condition uncertain coefficient



Case 2: Governing Equations

$$U_t + F(U)_x + G(U)_y = S(U)$$

Where:

$$U = (h, hv_x, hv_y)^T$$
$$F = (hv_x, hv_x^2 + 0.5k_{ap}g_z h^2, hv_x hv_y)^T$$
$$G = (hv_y, hv_x v_y, hv_y^2 + 0.5k_{ap}g_z h^2)^T$$

Case 2: Governing Equations Cont.

$$
\begin{aligned}
S &= (0, S_x, S_y)^T \\
S_x &= g_x h - \frac{V_x}{\sqrt{V_x^2 + V_y^2}} \left(g_z h + \frac{h V_x^2}{r_x} \right) \tan(\phi_{bed}) \\
&\quad - h k_{ap} \operatorname{sgn} \left(\frac{\partial V_x}{\partial y} \right) \frac{\partial (g_z h)}{\partial y} \sin(\phi_{int}) \\
S_y &= g_y h - \frac{V_y}{\sqrt{V_x^2 + V_y^2}} \left(g_z h + \frac{h V_y^2}{r_y} \right) \tan(\phi_{bed}) \\
&\quad - h k_{ap} \operatorname{sgn} \left(\frac{\partial V_y}{\partial x} \right) \frac{\partial (g_z h)}{\partial x} \sin(\phi_{int})
\end{aligned}
$$

Case 2: Result

For this case, we considered an inclined plane with the uncertain parameters of:

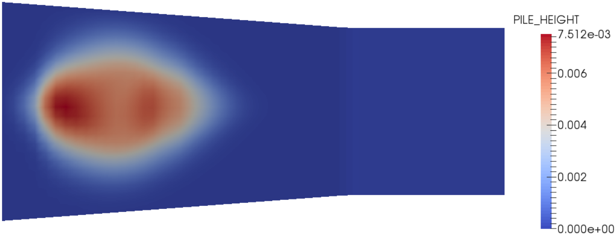
- Initial Volume $\pm 30\%$ *of* $2.64 \times 10^{-4} m^3$
- Bed Friction angle $(23^\circ, 40^\circ)$

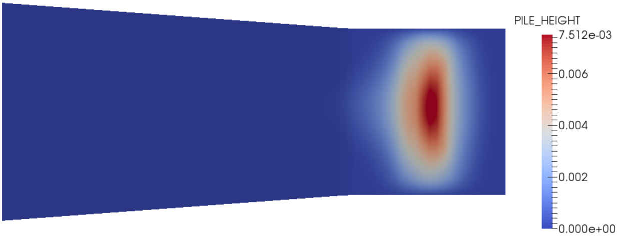
we simulated this process with 256 Latin Hyper cube samples. The functional that we are interested in this case is:

$$J = 0.5 \int_T \int_\Omega h^2 dX dt$$

Case 2: Result

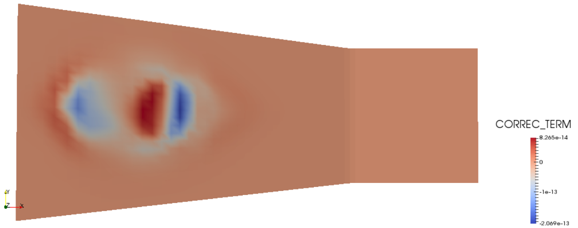
Pile height result for $t = 0.3$ sec and $t = 1$ sec

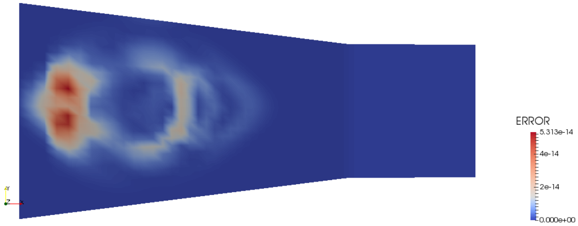




Case 2: Result

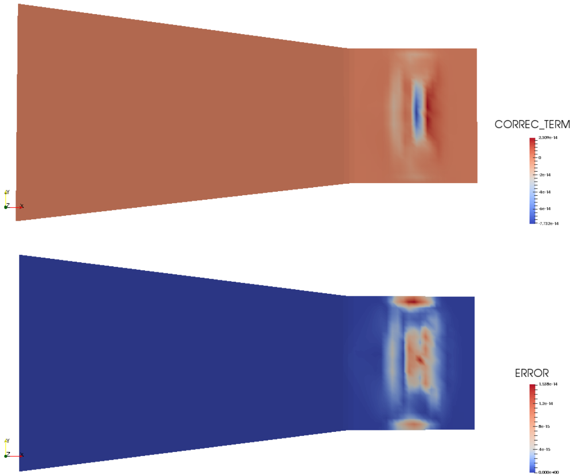
Correction term and Error for incline at $t = 0.3$ sec





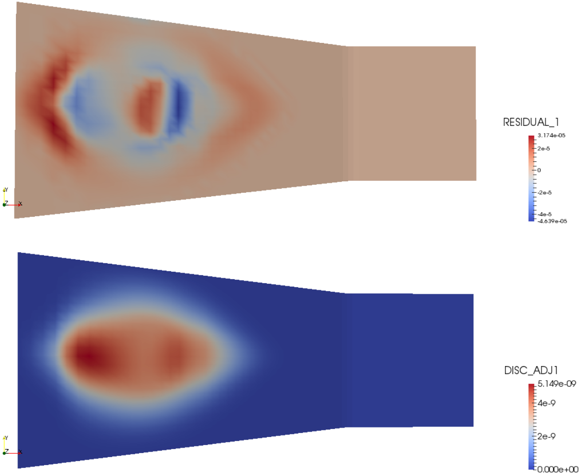
Case 2: Result

Correction term and Error for incline at $t = 1$ sec



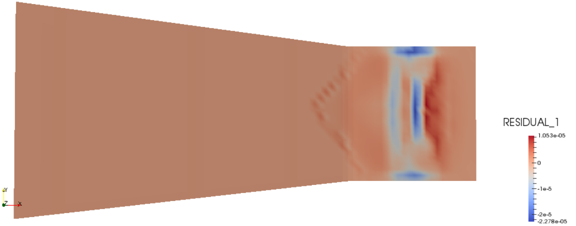
Case 2: Result

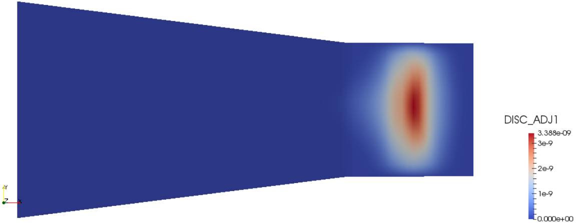
Residual and Adjoint result at $t = 0.3$ sec



Case 2: Result

Residual and Adjoint result at $t = 1$ sec





Summary

Simple approximations of residual and adjoints lead to a usable numerical error estimate that informs the surrogate construction.