Parameter Optimisation using POD: Application in Groundwater Flow

J. N. Wise

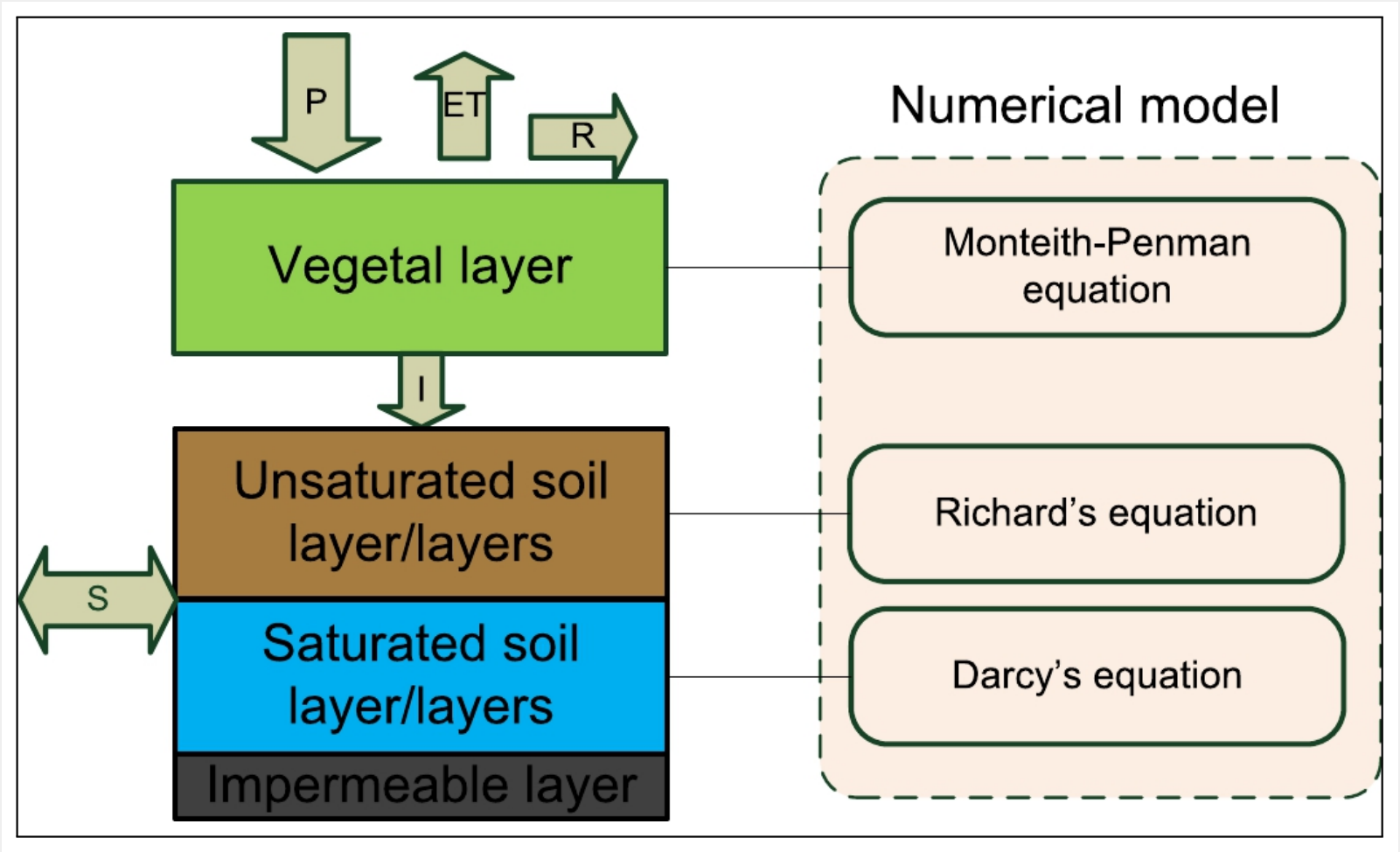
Supervisors: Prof. M. Batton-Hubert, Prof. G. Venter

Collaboration between EMSE and SU

December 7, 2012

Outline

Groundwater



Finite element equation and discretization

- Richard's equation: Describes head pressure h (in meters)

$$C(h)\frac{\partial h}{\partial t} = \nabla \cdot K(h)\nabla(h+z)$$

- Backward finite difference for time step

$$\frac{dh}{dt} = \frac{h - h^{old}}{\delta t}$$

- Variational formulation - multiply by test function $v(x,z) = v_1\varphi_1(x,z), ..., v_n\varphi_n(x,z)$

$$\underbrace{\int_\Omega vC\frac{h}{\delta t} + \int_\Omega \nabla v \cdot K\nabla h}_{\mathbf{Mh}} = \underbrace{\int_\Omega vC\frac{h^{old}}{\delta t} - \int_\Omega K\frac{\partial v}{\partial z} + \int_\Gamma vf}_{\mathbf{V}}$$

Richard equation: FE and equations

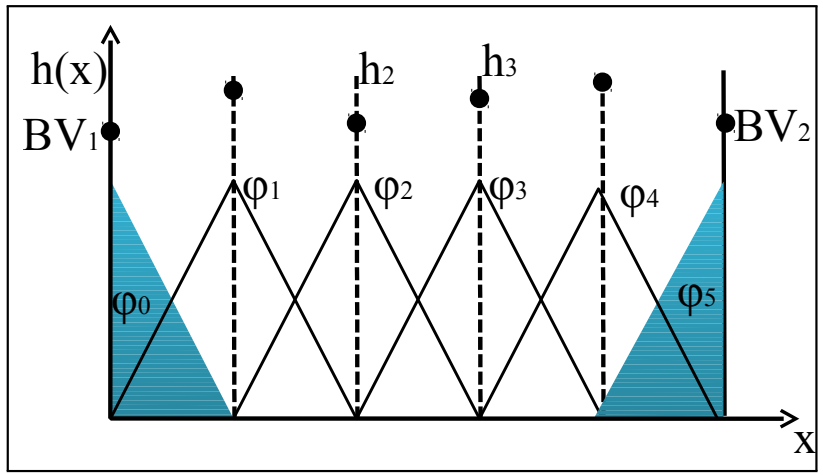
- Discretization

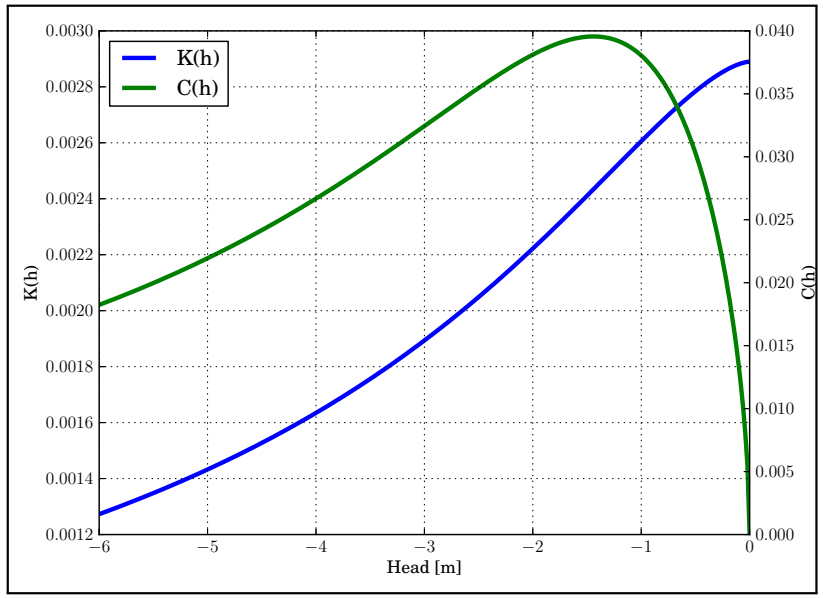
$$h(x,z,t) \approx \sum_{i=1}^{n} h_i(t)\varphi_i(x,z)$$

- Non-linear parameters

$$C(h) = \frac{d(\theta_s - \theta_r)(1 + (\frac{h}{h_g})^n)^{-m}}{dh}$$

$$K(h) = K_s \left(\frac{\theta(h) - \theta_r}{\theta_s - \theta_r} \right)^n$$





Matrix form

$$(\mathbf{M_1} + \mathbf{M_2})\,\mathbf{h} = \mathbf{V_1} + \mathbf{V_2} + \mathbf{V_3} \qquad [n \times n]$$

$$\left.\begin{aligned} \mathbf{M_1}(C) &= \frac{1}{\delta t}\int_\Omega vCv \\ \mathbf{M_2}(K) &= \int_\Omega \nabla v \cdot K \nabla v \end{aligned}\right\} \qquad [n \times n]$$

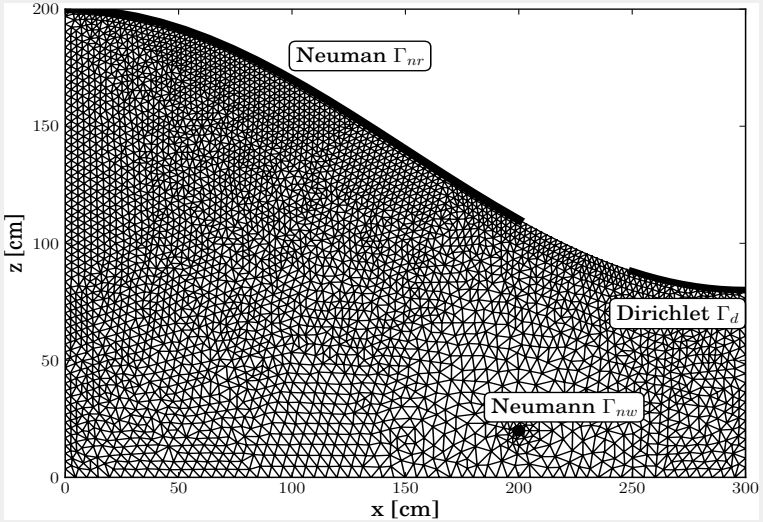
$$\left.\begin{aligned} \mathbf{V_1}(C) &= \frac{1}{\delta t}\int_\Omega Cv \\ \mathbf{V_2}(K) &= -\int_\Omega K\frac{\partial v}{\partial z} \\ \mathbf{V_3} &= \int_\Gamma vf(x,y,t) \end{aligned}\right\} \qquad [n \times 1]$$

POD

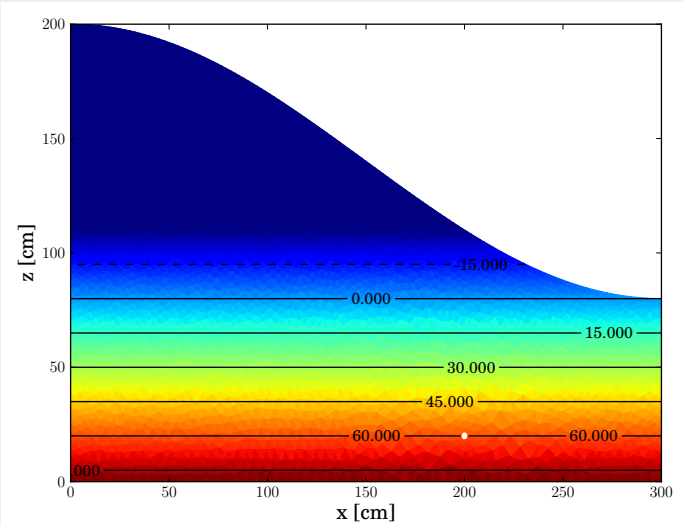
```
//----------//
// Matrices //
//----------//
varf m1(h, v) = int2d(Th)( Ch/dt * h * v);
varf m2(h, v) = int2d(Th)(Kh*dx(h)*dx(v)+Kh*dy(h)*dy(v))
                +on(7,h=(ymax -y+2*ampl));
matrix M1, M2;

//---------//
// Vectors //
//---------//
varf v1(h, v) = int2d(Th)( Ch/dt* hn*v);
varf v2(h, v) = int2d(Th)(-  Kh*dy(v));
varf vd(h, v) = on(7,h=(ymax -y+2*ampl));       // River
varf vnw(h, v) = int1d(Th,5)(-v*0.015);         // Pumping well
varf vnr(h, v) = int1d(Th,3)(v*BV*N.y*(x<200)); // Rainfall input

Vh V1, V2, Vd, Vnw, Vnr;
```

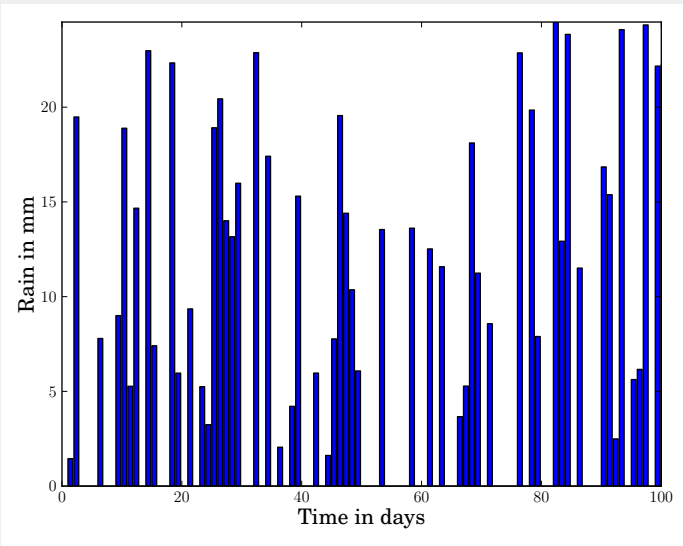
Boundary conditions

$$
\begin{aligned}
C(h)\frac{\partial h}{\partial t} &= \nabla \cdot K(h)\nabla(h+z) \\
h(x,z,t) &= (y-80)m \text{ on } \Gamma_d \\
K\nabla(h+z) &= -0.015 m/s \text{ on } \Gamma_{nw} \\
K\nabla(h+z) &= F_{rain}(t) m/s \text{ on } \Gamma_{nr} \\
\delta t = 24hr \quad & 0 \le t \le 100 days
\end{aligned}
$$



Initial condition and F_{rain}





Parameter optimisation problem

- Optimisation problem requires $\approx$ 100's simulations
- Need to reduce individual simulation times

Solution

- Reduce matrix size => number shape functions
- Run a number of full simulations (based on DOE)
- Capture the behaviour with global shape functions
- Global functions must optimally represent original response

PCA outline

- Collect snapshot set $\boldsymbol{X} = [h_1(x),\ h_2(x), ...,\ h_d(x)]$
- Optimisation problem: Find optimal global shape functions.

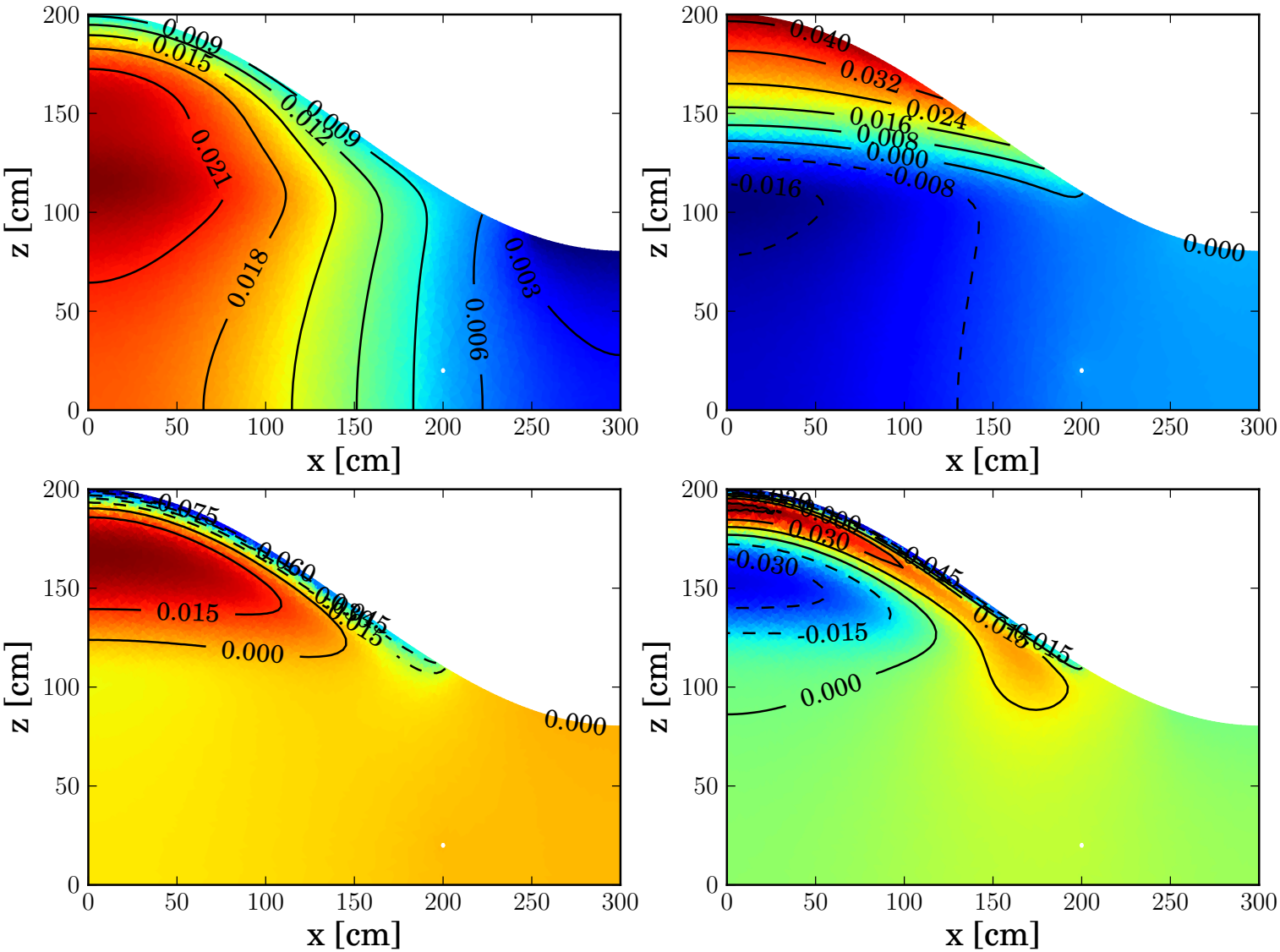
$$\text{Min}\|\boldsymbol{X} - proj_{\Psi_i}\boldsymbol{X}\|,\ \left\langle \Psi_i, \Psi_j \right\rangle = \delta_{ij},$$

- Set up covariance matrix

$$\boldsymbol{A} = \frac{1}{n}\boldsymbol{X}\boldsymbol{X}^T$$

- Turns out eigenvectors of $\boldsymbol{A}$ are optimal shape functions
- $\mathbf{U} = \{\Psi_1,\ \Psi_2,\ ...,\ \Psi_m\}$

First four eigenvectors



Reduced modelling with global shape functions

- Global shape function approximation

$$\hat{h}(x,z,t) = \sum_{j=1}^{m} \alpha_j(t)\Psi_j(x,z)$$

- Test function:

$$w(x,z) = w_1\Psi_1(x,z) + ... + w_m\Psi_m(x,z)$$

- Relation between $w(x,z)$ and $v(x,z)$:

$$\Psi_i(x,z) = \sum_{j=1}^{n} \Psi_{ij}\varphi_j(x,z)$$

- $\mathbf{U} = \{\Psi_1,\ \Psi_2,\ ...,\ \Psi_m\}$

Matrix form

$$\left(\hat{\mathbf{M}}_1 + \hat{\mathbf{M}}_2\right)\hat{\mathbf{h}} = \hat{\mathbf{V}}_1 + \hat{\mathbf{V}}_2 + \hat{\mathbf{V}}_3 \qquad [m \times m]$$

$$\left.\begin{aligned} \hat{\mathbf{M}}_1(C) &= \frac{1}{\delta t}\int_\Omega wCw \\ \hat{\mathbf{M}}_2(K) &= \int_\Omega \nabla w \cdot K \nabla w \end{aligned}\right\} \qquad [m \times m]$$

$$\left.\begin{aligned} \hat{\mathbf{V}}_1(C) &= \frac{1}{\delta t}\int_\Omega Cw \\ \hat{\mathbf{V}}_2(K) &= -\int_\Omega K\frac{\partial w}{\partial z} \\ \hat{\mathbf{V}}_3(f) &= \int_\Gamma wf(x,z,t) \end{aligned}\right\} \qquad [m \times 1]$$

Projection onto reduced basis

$$\left(\hat{\mathbf{M}}_1 + \hat{\mathbf{M}}_2\right)\hat{\mathbf{h}} = \hat{\mathbf{V}}_1 + \hat{\mathbf{V}}_2 + \hat{\mathbf{V}}_3 \qquad [m \times m]$$

$$\hat{\mathbf{M}}_1(C) = \mathbf{U}^{\mathbf{T}}\mathbf{M}_1(C)\mathbf{U}$$
$$\hat{\mathbf{M}}_2(K) = \mathbf{U}^{\mathbf{T}}\mathbf{M}_2(K)\mathbf{U}$$
$$\hat{\mathbf{V}}_1(C) = \mathbf{U}^{\mathbf{T}}\mathbf{V}_1(C)$$
$$\hat{\mathbf{V}}_2(K) = \mathbf{U}^{\mathbf{T}}\mathbf{V}_2(K)$$
$$\hat{\mathbf{V}}_3(f) = \mathbf{U}^{\mathbf{T}}\mathbf{V}_2(K)$$

To solve: $\overbrace{\mathbf{U}^{\mathbf{T}}(\mathbf{M_1} + \mathbf{M_2})\mathbf{U}}^{[m\times m]}\boldsymbol{\alpha} = \mathbf{U}^{\mathbf{T}}(\mathbf{V_1} + \mathbf{V_2} + \mathbf{V_3})$

$$\hat{\mathbf{M}}\boldsymbol{\alpha} = \hat{\mathbf{V}}$$
$$\hat{\mathbf{h}} = \mathbf{U}\boldsymbol{\alpha}$$

Linearize non-linear terms

- Linearize non-linear functions with PCA:

$$C \approx \widetilde{C} = \sum_{j=1}^{m^C} \alpha_j^C \psi_j^C(x,z), \qquad K \approx \widetilde{K} = \sum_{j=1}^{m^K} \alpha_j^K \psi_j^K(x,z)$$

- Find α^C and α^K with least squares projection: Matrices

$$\mathbf{Mh} => \widetilde{\mathbf{M}}\boldsymbol{\alpha} = \mathbf{U}^{\mathbf{T}} \left(\mathbf{M}_1(\widetilde{C}) + \mathbf{M}_2(\widetilde{K}) \right) \mathbf{U}\boldsymbol{\alpha} \qquad [m \times m \times m]$$

$$\mathbf{V} => \widetilde{\mathbf{V}} = \mathbf{U}^{\mathbf{T}} \left(\mathbf{V}_1(\widetilde{C}) + \mathbf{V}_2(\widetilde{K}) + \mathbf{V_3} \right) \qquad [m \times 1 \times m]$$

- Evaluate response

$$\widetilde{\mathbf{M}}\boldsymbol{\alpha} = \widetilde{\mathbf{V}}$$

$$\mathbf{h} = \mathbf{U}\boldsymbol{\alpha}$$

Error evaluation

- 'True' response $\boldsymbol{H}(\boldsymbol{P}^*) = \{\boldsymbol{h}_1^*, ..., \boldsymbol{h}_{100}^*\}$
- Error function (Vermeulen, 2004):

$$\textbf{RRMS} = 100 \cdot \frac{1}{nts} \sum_{i=1}^{nts} \sqrt{\frac{||\mathbf{h}_i(\boldsymbol{P}^*) - \hat{\mathbf{h}}_i(\boldsymbol{P}^*)||^2}{||\mathbf{h}_i(\boldsymbol{P}^*) - G_h||^2}}$$

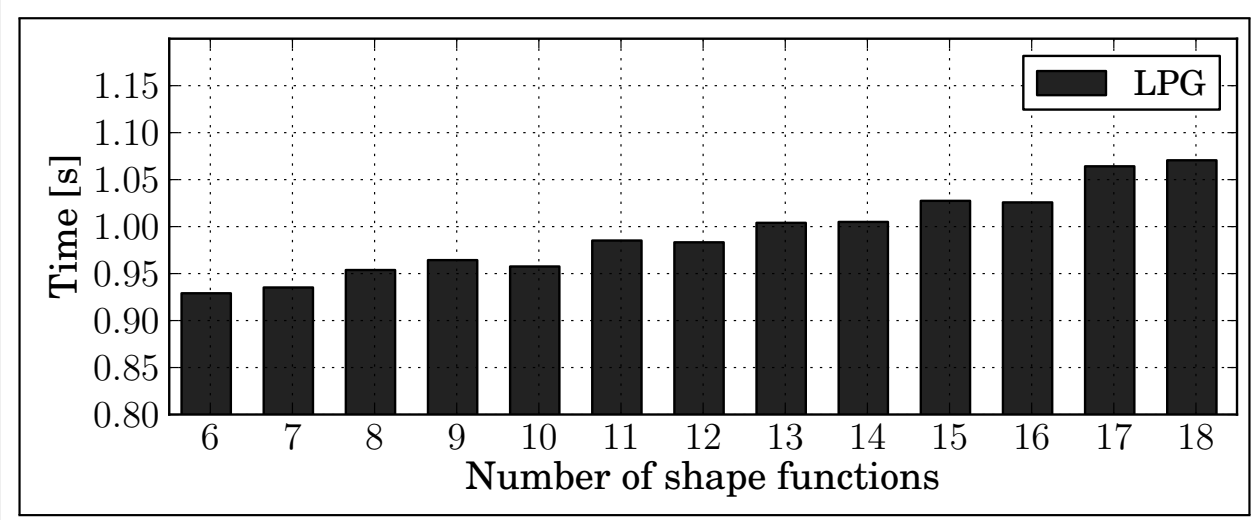
- Test LPG for parameter set $\boldsymbol{P}^*$:

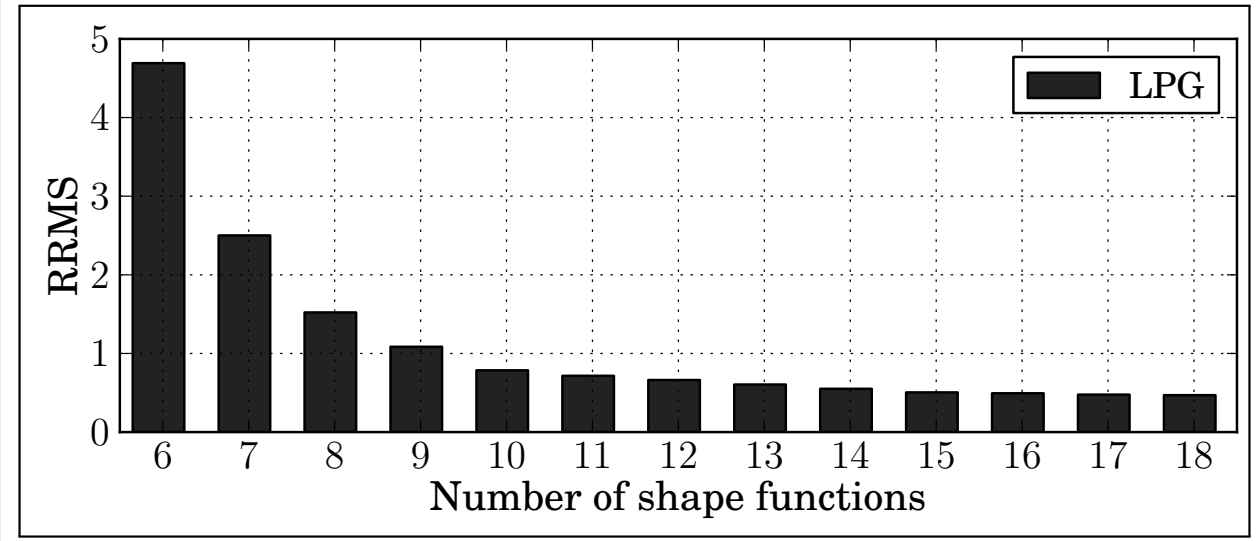
Table: Parameter set

$K_{s1}[cm/d]$	$K_{s2}[cm/d]$	$h_g[cm]$	m	θ_s	θ_r	l
0.032	0.048	29.2	0.4	0.34	0.0002	0.4

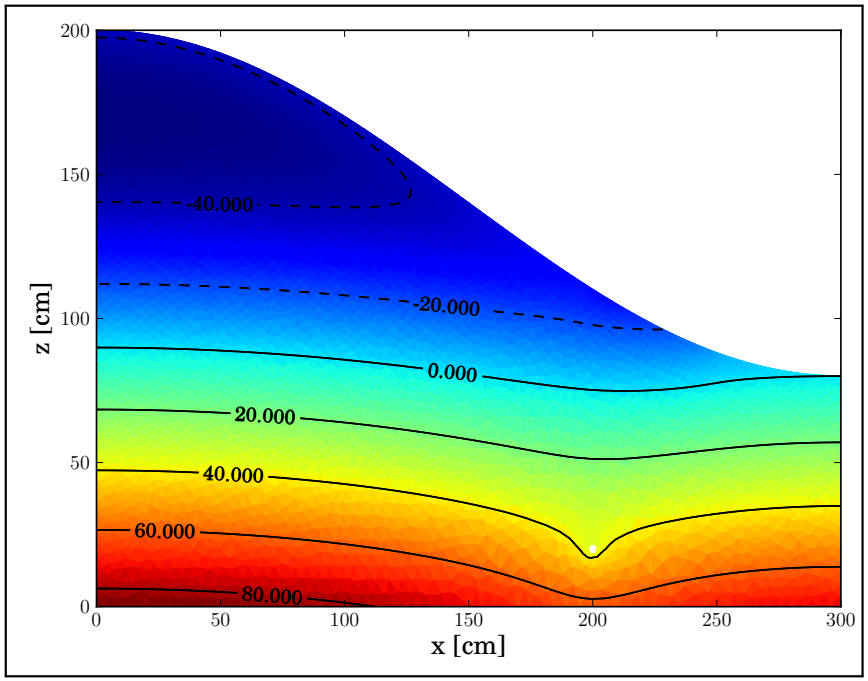
- Run full simulation, PCA, run reduced simulation

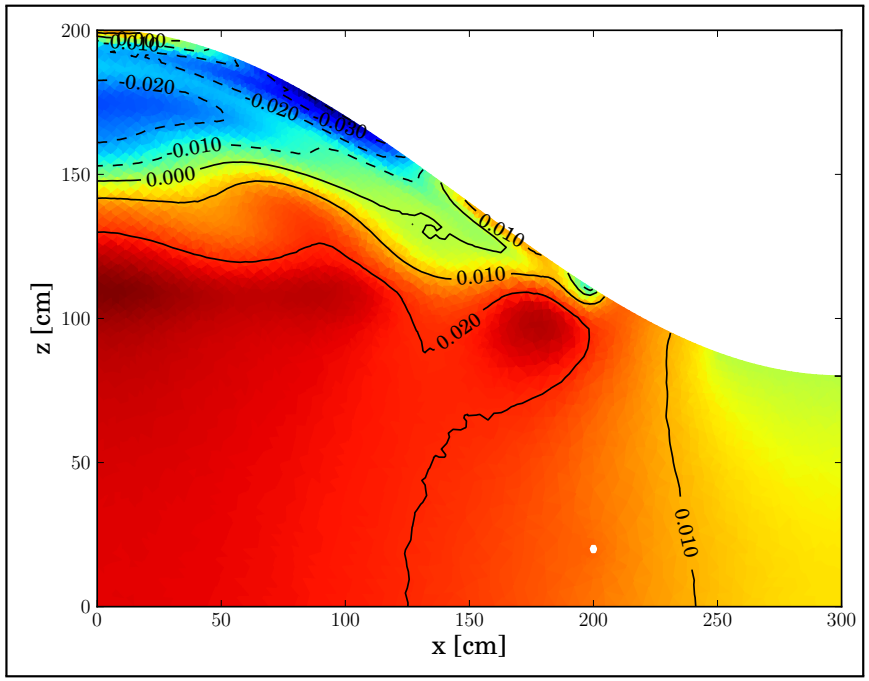
Error and time evaluations (FE simulation => 75 seconds)





True response and error with $m = 18$ and RRMS $= 0.45$

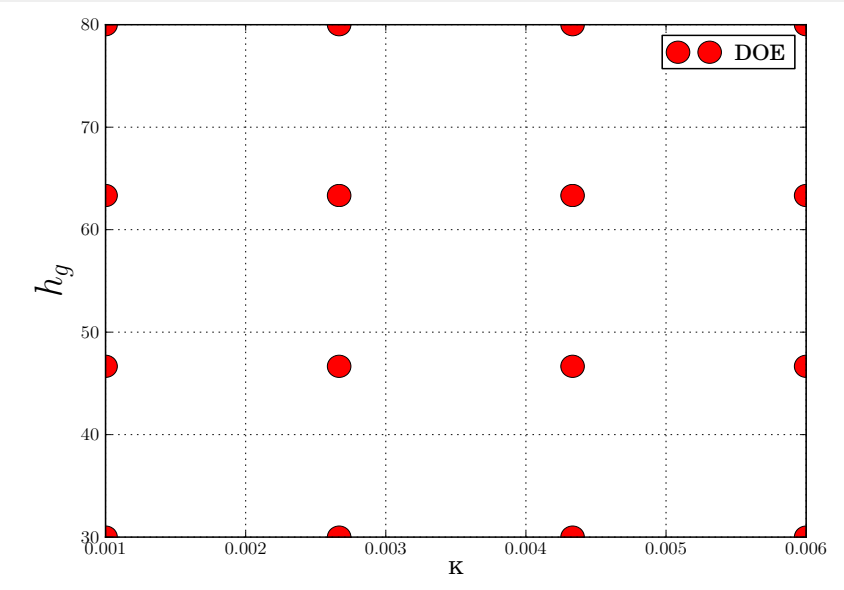


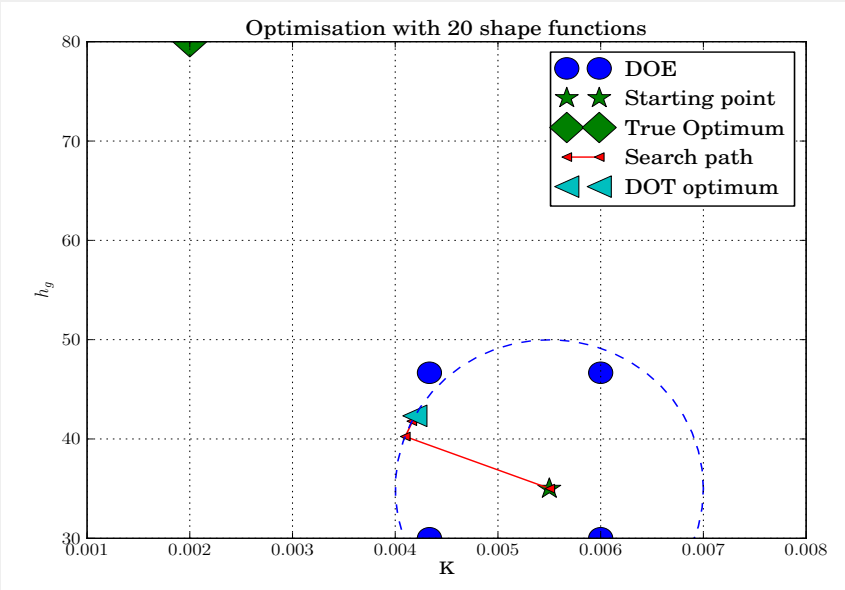


Simple optimisation problem:

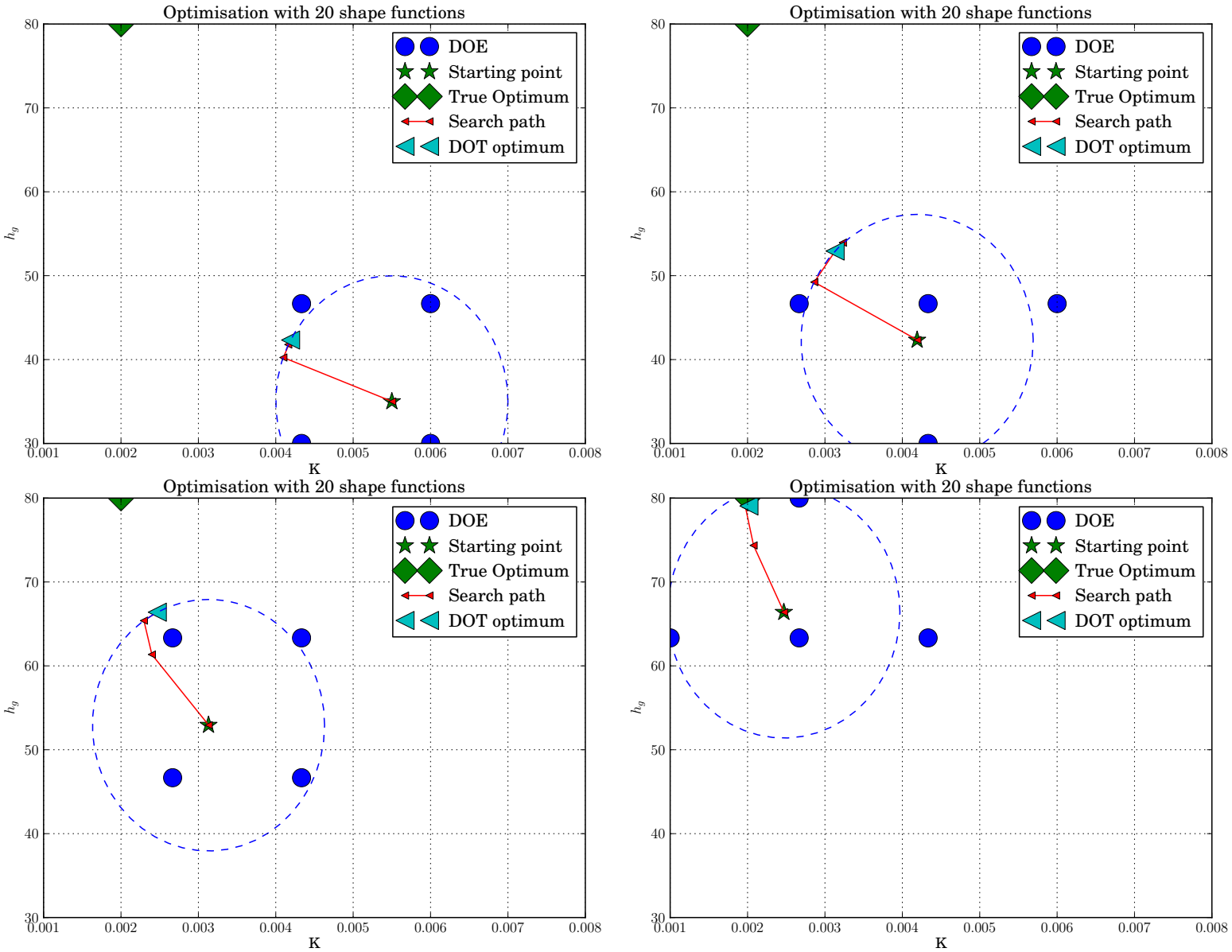
$$\begin{aligned} \text{Minimize :} \quad & F(\mathbf{P}) = \mathbf{RRMS}(\mathbf{H}(\mathbf{P}^*), \hat{\mathbf{H}}(\mathbf{P})) \\ \text{Subject to :} \quad & \boldsymbol{P} \geq \boldsymbol{P_l} \\ & \boldsymbol{P} \leq \boldsymbol{P_u} \end{aligned}$$

Design of experiments to capture different behaviour

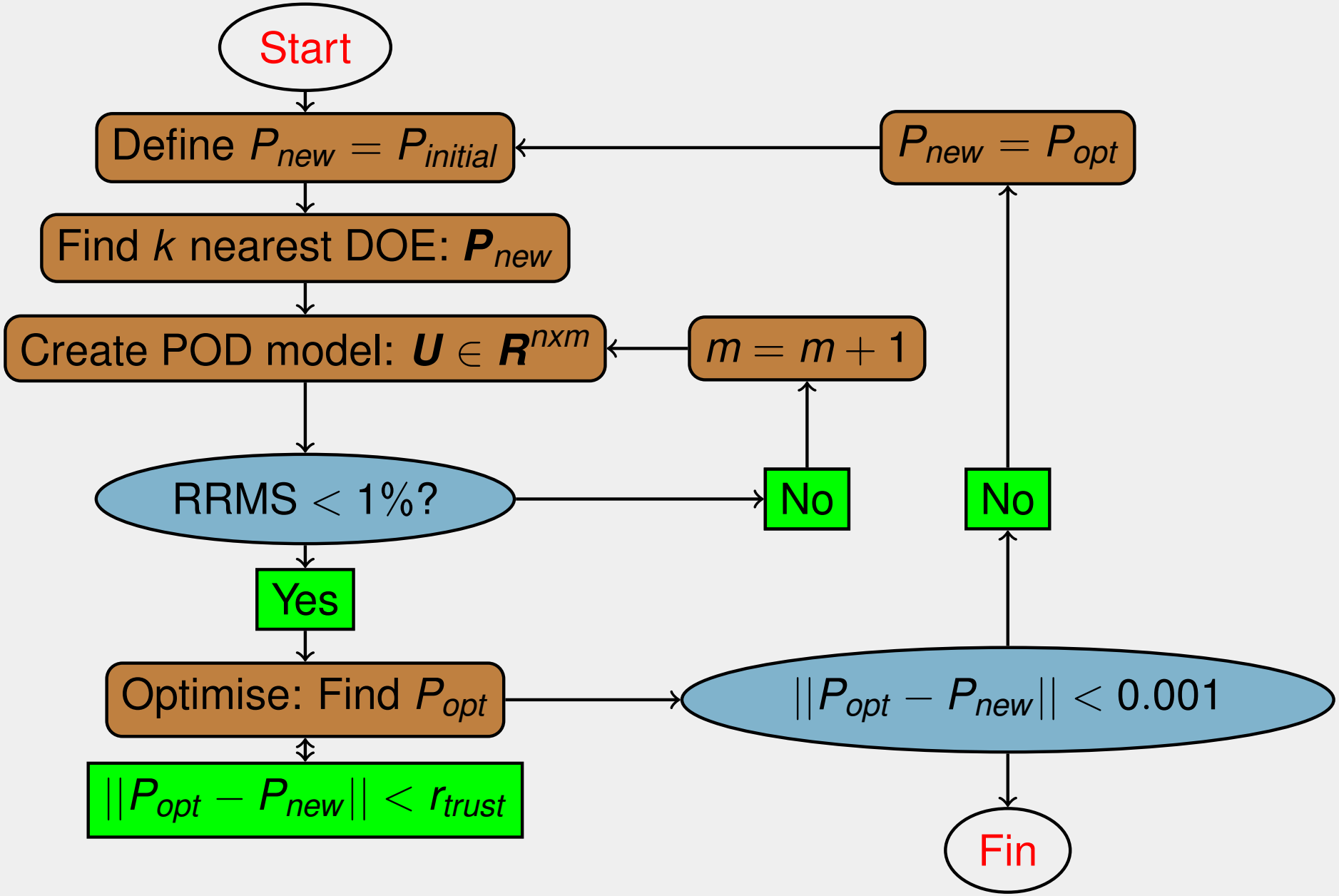




POD Trust region (Fahl:2000), (Alexandrov:1998)



Algorithm


Start
Define $P_{new} = P_{initial}$
Find k nearest DOE: $\boldsymbol{P}_{new}$
Create POD model: $\boldsymbol{U} \in \boldsymbol{R}^{nxm}$
$m = m + 1$
RRMS $< 1\%$?
No
Yes
Optimise: Find P_{opt}
$||P_{opt} - P_{new}|| < r_{trust}$
$P_{new} = P_{opt}$
No
$||P_{opt} - P_{new}|| < 0.001$
Fin

Results from 7 variable optimisation with $m = 15$ and $K = 3$

P:	Ks1	Ks2	hg	m	thetas	thetar	infilt
Init	0.04	0.056	12.133	0.43	0.36	0.0006	0.4666
Fin	0.033	0.0486	29.508	0.4112	0.3626	0.0006	0.3879
True	0.032	0.048	29.2	0.4033	0.34	0.0002	0.4

Time considerations with scaled trust radius $r_{trust} = 0.3$

- Needs ≈ 100 iterations per trust region
- Run's four cycles before convergence
- Total function calls ≈ 400
- Each full FE function call ≈ 75s
- Each POD call ≈ 1s
- Therefore speedup of $75x$

Conclusion

- Very useful having FreeFem++ to create matrices
- POD method show excellent potential in time reduction
- Good potential in extrapolation
- Useful for parameter optimisation
- Yields information over whole domain unlike metamodelling techniques