SAND2008-3199C

Biofilm Characterization using Large Scale Optimization

Bart van Bloemen Waanders
Sandia National Laboratories

Judith Hill
Oakridge National Laboratories

Kevin Long
Texas Tech University

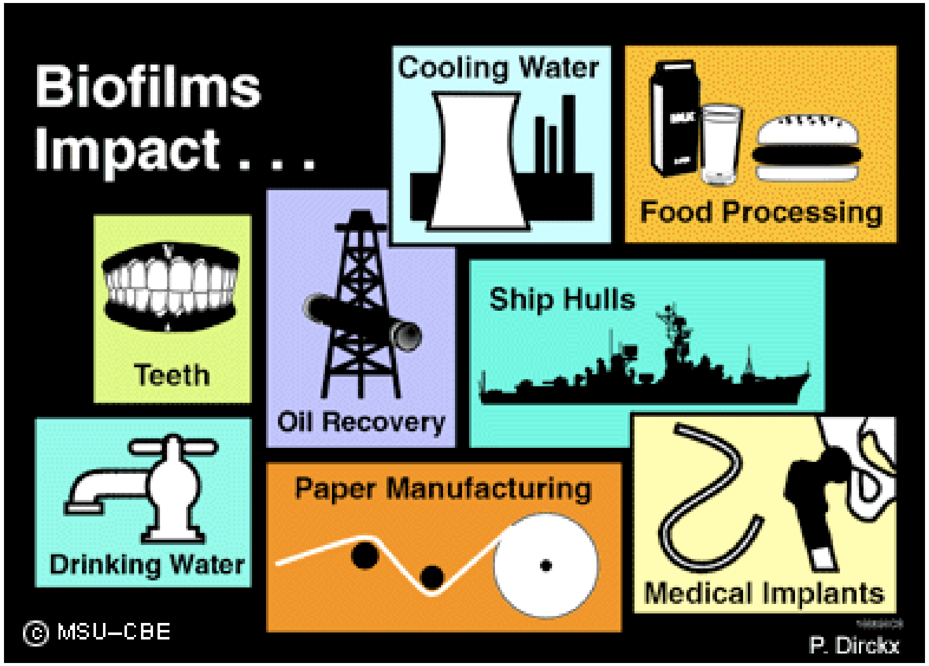
SIAM Optimization Conference
May, 2008

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

Sandia National Laboratories

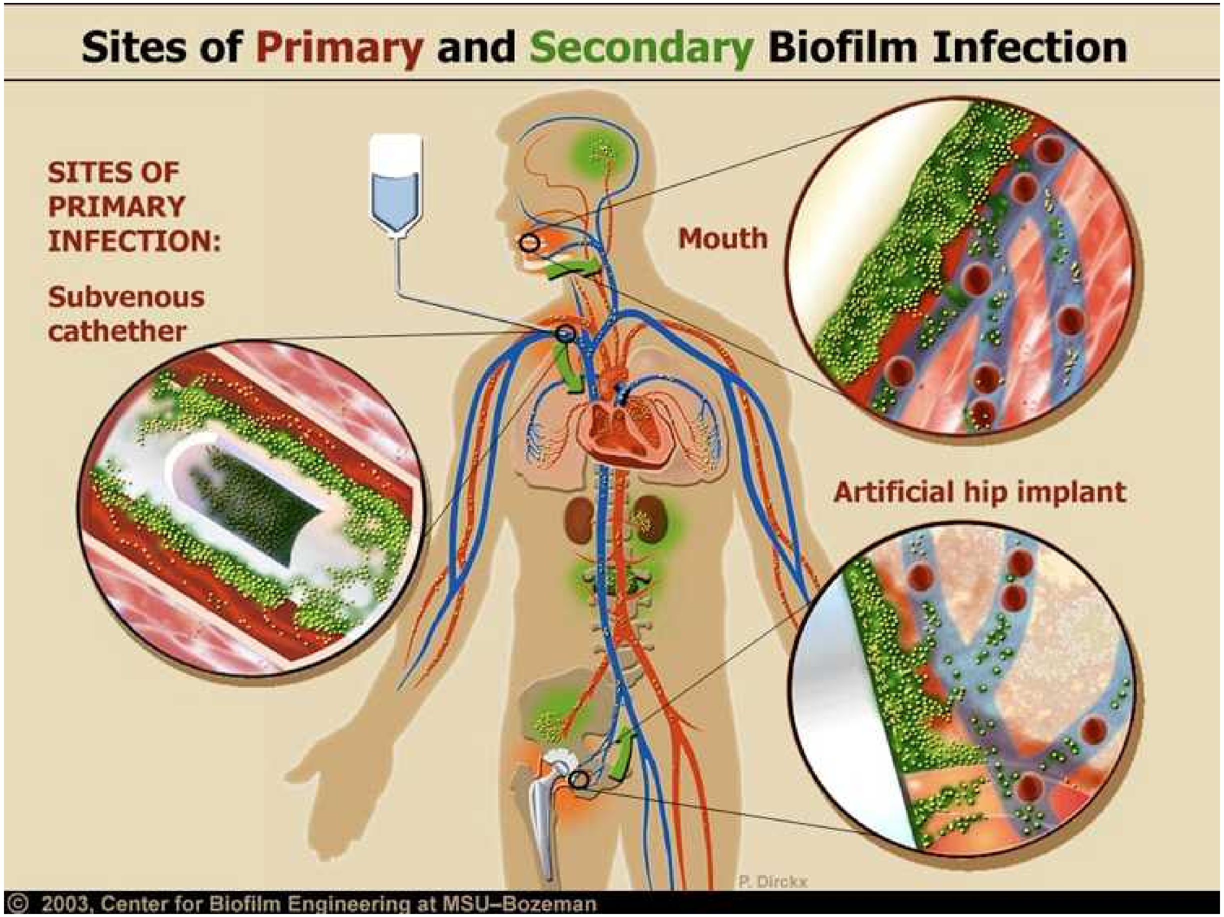
Motivation: Definition and Impact

A biofilm is a complex aggregation of microorganisms marked by the excretion of a protective and adhesive matrix.

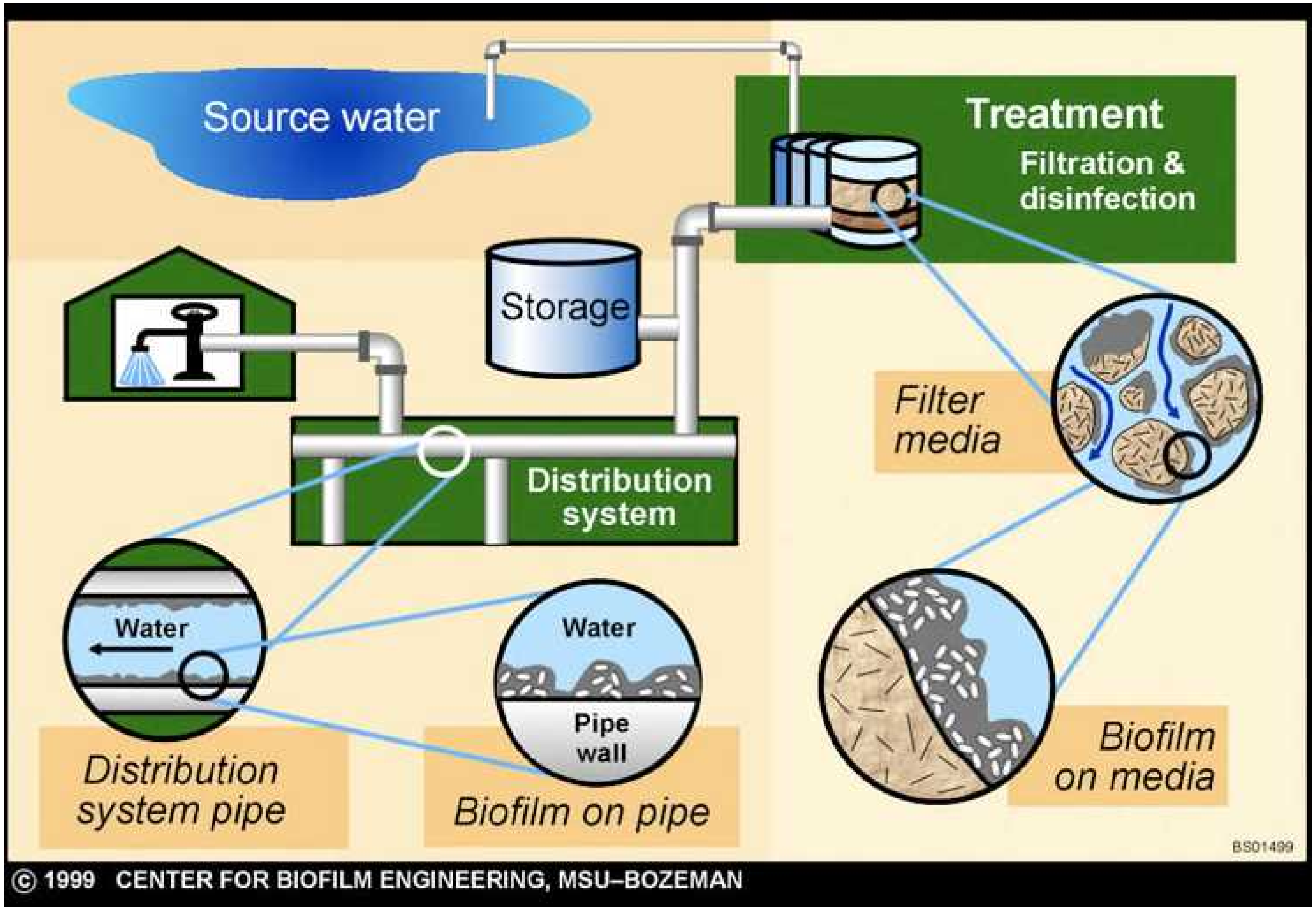


Sandia National Laboratories

Motivation: Human Physiology



Motivation: Water Distribution & Security



Sandia National Laboratories

Goals of the project:

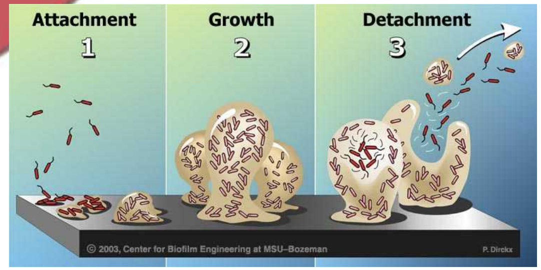
- Develop simulation capabilities to predict the dynamics of biofilms
- Calibrate model with experimental observations
- Develop efficient and flexible extension capabilities
- Characterize uncertainties

Outline

- Motivation
- Physics
- Inversion
- Implementation
- Numerical examples/results

Goal of the presentation: 1) motivate **PDE C**onstrained **O**ptimization through biofilm problem, 2) demonstrate convenient PDECO framework

- Bacteria Adhesion
- EPS Production
- Bacteria duplication
- Transport of interface
- Detachment



Biofilm Physics

Multi-species Model for growth:

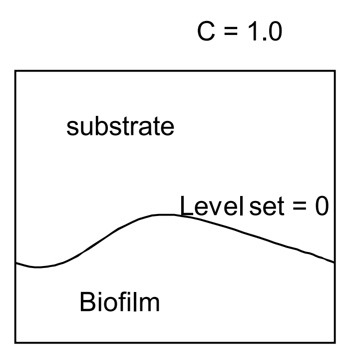
(Alpkvist and Klapper 06)

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

Simplified model:

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

Computational domain:

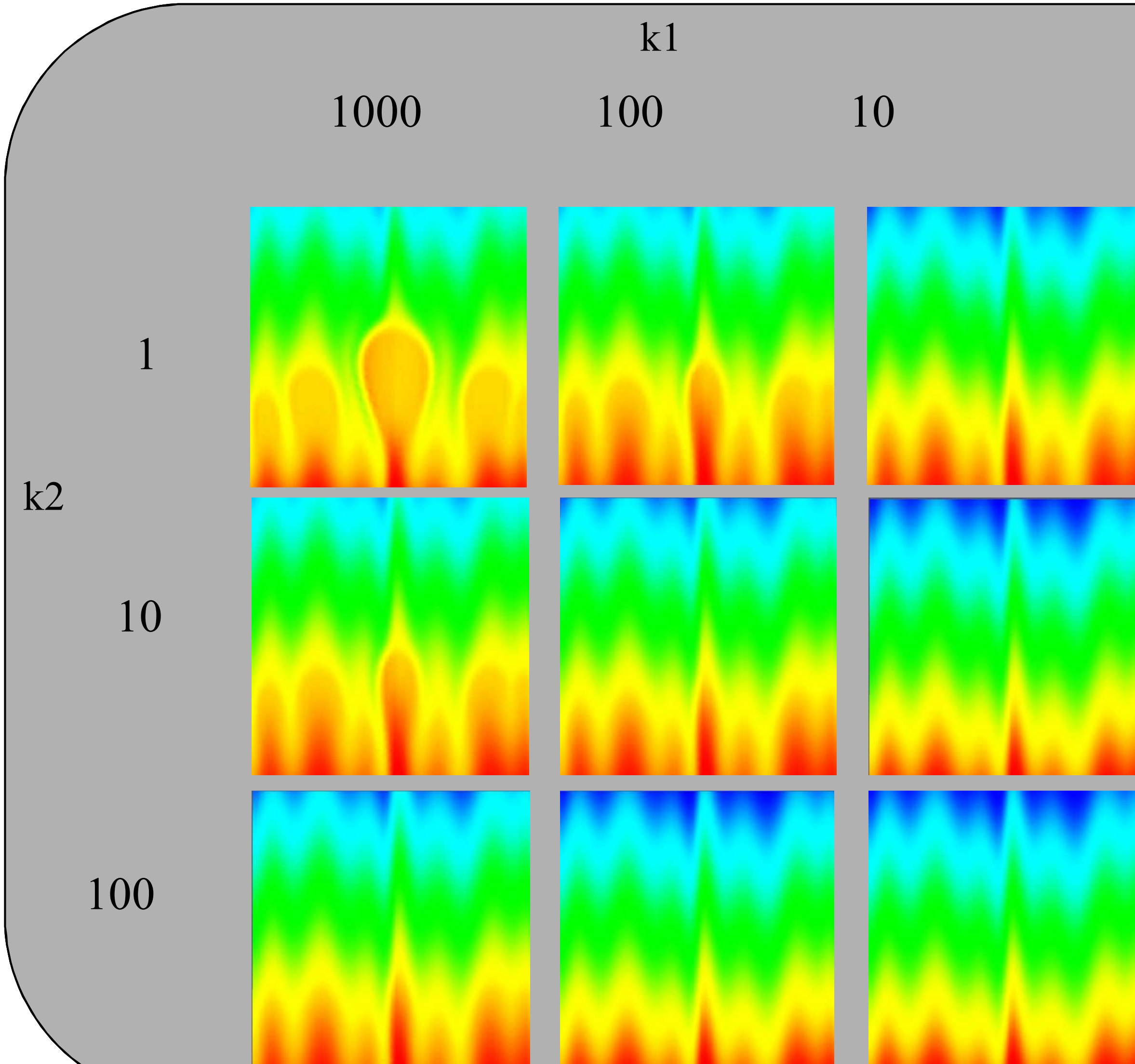


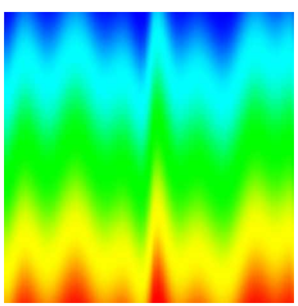
Forward Simulation Results
(Realizations shown at final timestep)

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

<u>Simulation Details</u>:

- deltaT = 0.005
- grid 64x64
- D = 1.0
- Newton-GMRES
- ILU preconditioner
- CPU time = 70s





Initial Condition

Forward Simulation Results

(Realizations shown at final timestep)

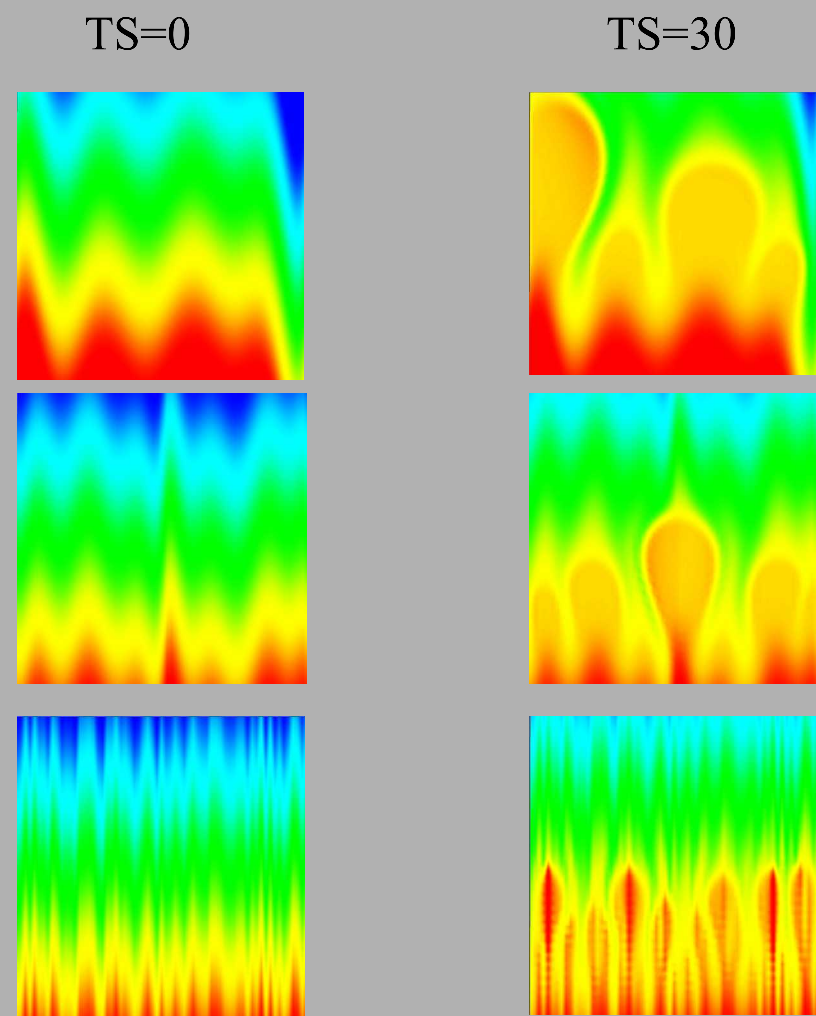
QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

Simulation Details:

- timesteps = 30
- deltaT = 0.005
- grid 64x64
- D = 1.0
- Newton-GMRES
- ILU preconditioner
- CPU time = 70s

Constant:

- K1 = 1000.0
- K2 = 1.0



Sandia National Laboratories

Biofilm Optimization Formulation

The PDE constrained optimization problem

$$\min_{\rho,\rho_o} \; \mathcal{F}(\rho,\rho_u) = \frac{1}{2}\sum_{i=0}^{N}\int_0^T \int_\Omega (\rho-\rho^*)^2\delta(x-x_i)\,\mathbf{dx}\,\mathbf{dt} + \frac{\beta}{\mathbf{2}}\int_{\mathbf{\Omega}} \rho_{\mathbf{0}}^{\mathbf{2}}\,\mathbf{dx}$$

subject to:

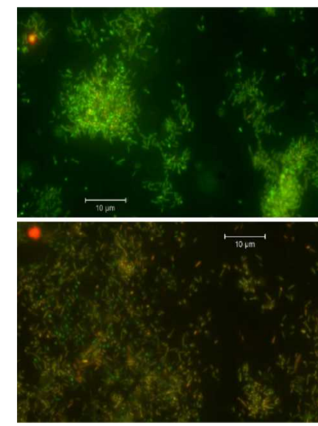
$$\frac{\partial\rho}{\partial t} + \frac{D}{\rho_b}\nabla c\cdot\nabla\rho = 0 \qquad \text{in} \quad \Omega\times[0,T]$$

$$\rho = \rho_0 \qquad \text{in} \quad \Omega\times\{t=0\}.$$

$$D\Delta c = \frac{k_1 c\rho}{k_2 + c} \qquad \text{in} \quad \Omega$$

$$c = c_D \qquad \text{in} \quad \Gamma_D$$

Match confocal microscope images



Biofilm Optimality Conditions

The forward coupled problem

$$D\Delta c = \frac{k_1 c\rho}{k_2 + c} \quad \text{in } \Omega$$

$$c = c_D \quad \text{in } \Gamma_D$$

$$\frac{\partial \rho}{\partial t} + \frac{D}{\rho_b}\nabla c \cdot \nabla \rho = 0 \quad \text{in } \Omega \times [0, T]$$

$$\rho = \rho_0 \quad \text{in } \Omega \times \{t = 0\}.$$

The adjoint problem

$$D\Delta\lambda_1 - \frac{k_1 k_2 \rho}{(k_2 + c)^2}\lambda_1 - \frac{D}{\rho_b}\lambda_2 \Delta\rho = 0 \quad \text{in } \Omega$$

$$-D\nabla\lambda_1 \cdot n + \lambda_2 \frac{D}{\rho_b}\nabla\rho \cdot n = 0 \quad \text{on } \Gamma_N$$

$$\lambda_1 = 0 \quad \text{on } \Gamma_D$$

$$-\frac{\partial \lambda_2}{\partial t} - \frac{D}{\rho_b}\lambda_2 \Delta c - \frac{k_1 c \lambda_1}{k_2 + c} = -\Sigma(\rho - \rho^*)\delta_j \quad \text{in } \Omega \times [0, T]$$

$$\lambda_2 = 0 \quad \text{in } \Omega \times \{t = 0\}$$

The initial concentration equation

$$\beta\,\rho_0 - \lambda_2|_{t=0} = 0 \quad \text{in } \Omega.$$

Implementation Goals

Need **efficient** capability to implement **embedded** algorithms (i.e. PDECO)

Possible solutions: compile time transformation (**tricky** on complex codes), friendly interfaces, i.e. Comsol (**slow, inflexible**), armies of programmers (**expensive, error-prone, unmaintainable**

Our approach: 1) State in **mathematical form** the general problem of "writing" an efficient intrusive code 2) Write (**by hand**) a code to solve that meta-problem, 3) Write papers while the computer does the dirty work

Added benefits: The key to intrusion turns out to **simplify automation** as well a mathematical definition of the simulation development process aids in establishing **code correctness**

Sandia National Laboratories

Implementation Mathematics

- Test/Unkown/Auxiliary variables
- Functional spaces
- Frechet differentiation
- Discrete spaces

Examples:

Steady Navier-Stokes flow

$$F[\mathbf{v},\mathbf{u},q,p] = \int_\Omega [\nu \nabla \mathbf{v} : \nabla \mathbf{u} + p \nabla \cdot \mathbf{v} + \mathbf{v} \cdot (\mathbf{u} \cdot \nabla) \mathbf{u} + q \nabla \cdot \mathbf{u}]$$

Lagrangian for Poisson source inversion w/ Tikhonov

$$F[v,\mu,\beta,u,\lambda,\alpha] = \int_\Omega [(u-u^*)\, v + \nabla\lambda \cdot \nabla v]\, dx + \\ + \int_\Omega [\nabla\mu \cdot \nabla u + \mu\alpha]\, dx + \\ + \int_\Omega [\nabla\alpha \cdot \nabla\beta + \lambda\beta]\, dx$$

Approach unifies diverse set of problems:

A linearized forward problem: find Newton step $\delta\mathbf{u}$ at $\mathbf{u}_0$

$$\left.\frac{\partial \hat{F}}{\partial v_i}\right|_{\mathbf{u}_0} + \left.\frac{\partial^2 \hat{F}}{\partial v_i \partial u_j}\right|_{\mathbf{u}_0} \delta u_j = 0$$

A quadratic eigensystem: find σ, $\mathbf{u}$

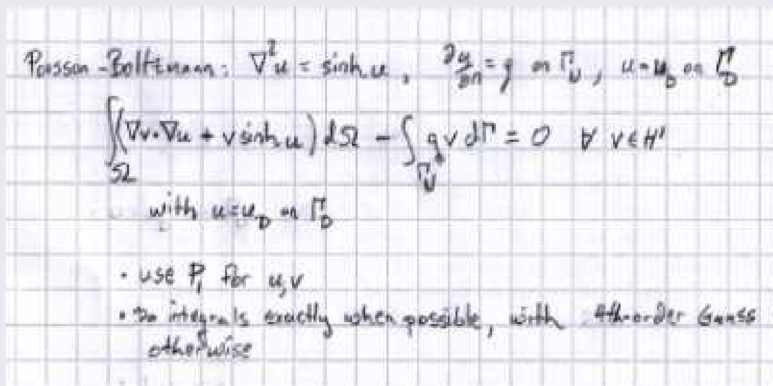
$$\left.\frac{\partial^2 \hat{F}}{\partial v_i \partial u_j}\right|_0 u_j + \left.\frac{\partial^3 \hat{F}}{\partial v_i \partial u_j \partial \sigma}\right|_0 \sigma u_j + \frac{1}{2} \left.\frac{\partial^4 \hat{F}}{\partial v_i\, \partial u_j\, \partial \sigma^2}\right|_0 u_j \sigma^2 = 0$$

A sensitivity problem: find $\partial\mathbf{u}/\partial\sigma$ at $\mathbf{u}_0$

$$\left.\frac{\partial^2 \hat{F}}{\partial v_i\, \partial \sigma}\right|_{\mathbf{u}_0} + \left.\frac{\partial^2 \hat{F}}{\partial v_i\, \partial u_j}\right|_{\mathbf{u}_0} \frac{\partial u_j}{\partial \sigma} = 0$$

Implementation Syntax

Sundance is a toolkit that is based on these principles



Poisson-Boltzmann: $\nabla^2 u = \sinh u$, $\frac{\partial u}{\partial n} = g$ on Γ_N, $u = u_D$ on Γ_D

$$\int_\Omega (\nabla v \cdot \nabla u + v \sinh u)\, d\Omega - \int_{\Gamma_N} g v \, d\Gamma = 0 \quad \forall\, v \in H^1$$

with $u = u_D$ on Γ_D

- use P_1 for u, v
- Do integrals exactly when possible, with 4th-order Gauss otherwise

Poisson-Boltzmann equation in a notebook

```
u = UnknownFunction(Lagrange(1))
v = TestFunction(Lagrange(1))

quad = GaussianQuadrature(4)

weak = Integral(omega,(grad*u)*(grad*v)+v*sinh(u), quad)
```

Poisson-Boltzmann equation in Sundance

Implementation Syntax for Optimization: Approach 1

The forward coupled problem

$$
\begin{aligned}
D\Delta c &= \frac{k_1 c\rho}{k_2+c} && \text{in } \Omega\\
c &= c_D && \text{in } \Gamma_D\\
\frac{\partial\rho}{\partial t} + \frac{D}{\rho_b}\nabla c\cdot\nabla\rho &= 0 && \text{in } \Omega\times[0,T]\\
\rho &= \rho_0 && \text{in } \Omega\times\{t=0\}.
\end{aligned}
$$

The adjoint problem

$$
\begin{aligned}
D\Delta\lambda_1 - \frac{k_1k_2\rho}{(k_2+c)^2}\lambda_1 - \frac{D}{\rho_b}\lambda_2\Delta\rho &= 0 && \text{in } \Omega\\
-D\nabla\lambda_1\cdot n + \lambda_2\frac{D}{\rho_b}\nabla\rho\cdot n &= 0 && \text{on } \Gamma_N\\
\lambda_1 &= 0 && \text{on } \Gamma_D\\
-\frac{\partial\lambda_2}{\partial t} - \frac{D}{\rho_b}\lambda_2\Delta c - \frac{k_1 c\lambda_1}{k_2+c} &= -\Sigma(\rho-\rho^*)\delta_j && \text{in } \Omega\times[0,T]\\
\lambda_2 &= 0 && \text{in } \Omega\times\{t=0\}
\end{aligned}
$$

The initial concentration equation

$$\beta\,\rho_0 - \lambda_2|_{t=0} = 0 \qquad \text{in } \Omega.$$

forward

```
Expr state = Integral(interior, (u - u0)*uHat* 1.0/deltaT + v*(grad*u)*uHat
                                + D*(grad*uHat)*(grad*u), q2);
Expr stateBC = EssentialBC(left, u*uHat, q2);

stateProb = rcp(new LinearProblem(mesh_, state, stateBC, mu, u, vecType));
```

adjoint

```
CellFilter sensors = sensorFilter() ;

Expr adjoint = Integral(interior, (lambda - lambda0)*v/deltaT
                                    + D*(grad*u)*(grad*lambda)
                                    + v*(grad*u)*lambda, q2)
                                    + Integral(sensors, v*(u - uTarget), q2) ;

Expr adjointBC = EssentialBC(left, lambda*v, q2)
               + EssentialBC(bottom, lambda*v, q2) ;

adjointProb = rcp(new LinearProblem(mesh_, adjoint, adjointBC,
                                     v, lambda, vecType));
```

sensitivity

```
Expr sens = Integral(interior_, -Reg_ * u0_*beta + lambda0_*beta + beta*u, q2);

Expr sensBC;
Expr sensProb = new LinearProblem(mesh, sens, sensBC, beta, u, vecType);
```

Sandia National Laboratories

Implementation Syntax for Optimization: Approach 2

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

```
Expr lag = Integral(interior, 0.5*pow(u - target, 2.0) *direcDelta + R*alpha*alpha
                    + (u - uOld)*lambda + deltaT*D*(grad*lambda)*(grad*u)
                    + lambda*v*(grad * u), q2);
```

Lagrangian definition

```
Expr lagBC = EssentialBC(left, lambda*u, q2);
```

Boundary conditions

```
Functional L(mesh, lag, lagBC, vecType);
LinearProblem stateProb = L.linearVariationalProb(lambda, lambda0,
                                                  u, alpha, alpha0);
```

Define functional

Variational statement

```
LinearSolver<double> solver
  = LinearSolverBuilder::createSolver(solverParams);
```

Solver definition

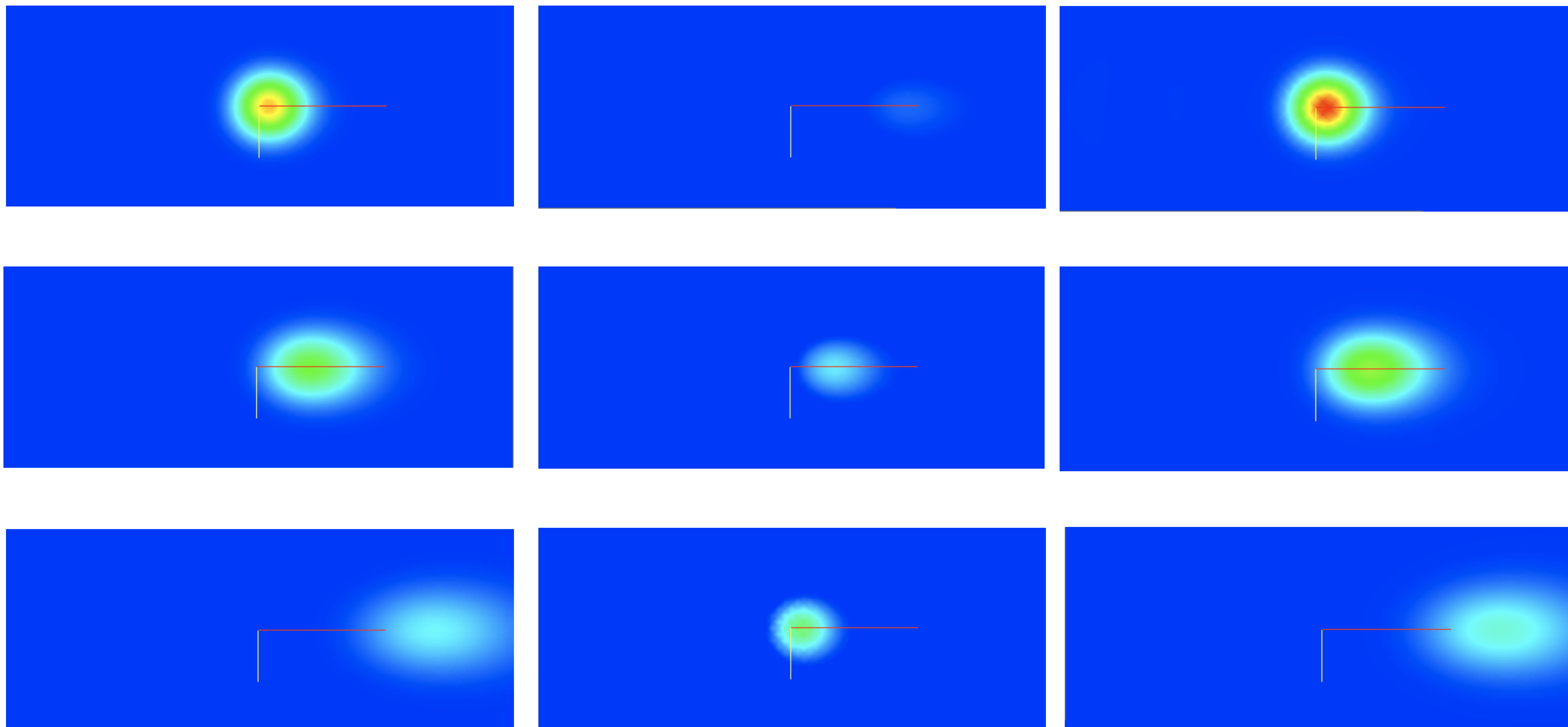
```
  for (int i=0; i<Nts; i++)
 {

  uSoln = stateProb.solve(solver);
  CopyOldSol(uSoln, uOld);

 }
```

Time-stepping loop

Sandia National Laboratories

Example: Convection Diffusion Forward and Adjoint

Target : Adjoint : Inversion:



Sandia National Laboratories

Biofilm Inversion Formulation and implementation
Approach 1

The PDE constrained optimization problem

$$\min_{\rho,\rho_o} \mathcal{F}(\rho,\rho_u) = \frac{1}{2}\sum_{i=0}^{N}\int_0^T \int_\Omega (\rho-\rho^*)^2\delta(x-x_i)\,\mathbf{dx}\,\mathbf{dt} + \frac{\beta}{\mathbf{2}}\int_\Omega \rho_0^2\,\mathbf{dx}$$

subject to:

$$\begin{aligned} \frac{\partial\rho}{\partial t} + \frac{D}{\rho_b}\nabla c\cdot\nabla\rho &= 0 && \text{in } \Omega\times[0,T] \\ \rho &= \rho_0 && \text{in } \Omega\times\{t=0\}. \\ D\Delta c &= \frac{k_1 c\rho}{k_2+c} && \text{in } \Omega \\ c &= c_D && \text{in } \Gamma_D \end{aligned}$$

Additional complications:

- SUPG
- Crank-Nicholson

```
Expr state = Integral(interior_, (rho - rho0_)*rhoHat/deltaT
                                 +0.5*(D_/rhoB_)*((grad*rho)*(grad*c))*rhoHat
                                 +0.5*(D_/rhoB_)*((grad*rho0_)*(grad*c0_))*rhoHat

                                 +(rho - rho0_)/deltaT
                                   *stau*(D_/rhoB_)*((grad*st2)*(grad*rhoHat))
                                 +0.5*(D_/rhoB_)*((grad*rho)*(grad*c))*rhoHat
                                   *stau*(D_/rhoB_)*((grad*st2)*(grad*rhoHat))
                                 +0.5*(D_/rhoB_)*((grad*rho0_)*(grad*c0_))*rhoHat
                                   *stau*(D_/rhoB_)*((grad*st2)*(grad*rhoHat))

                                 -D_*(grad*c)*(grad*cHat)
                                 -Hrho*k1_*c/(k2_+c) * c, q2);
```

```
Expr adjoint1 = Integral(interior_, (lrho - lrho1_ )*lrhoHat/deltaT *deltaT

                                 +0.5*(D_/rhoB_)*((grad*lrhoHat)*(grad*c0_))*lrho*deltaT
                                 -0.5*k1_*dHlrho*c0_/(k2_+c0_)*lc*lrhoHat

                                 +0.5*(D_/rhoB_)*((grad*lrhoHat)*(grad*c1_))*lrho1_ *deltaT
                                 -0.5*k1_*dHlrho1*c1_/(k2_+c1_)*lc*lrhoHat

                                 +(lrho - lrho1_ )/deltaT
                                   *atau*(D_/rhoB_)*((grad*at2)*(grad*lrhoHat))
                                 +0.5*(D_/rhoB_)*((grad*lrho)*(grad*c0_))*lrho
                                   *atau*(D_/rhoB_)*((grad*at2)*(grad*lrhoHat))
                                 -0.5*k1*dHlrho*c0_/(k2_+c0_)*lc
                                   *atau*(D_/rhoB_)*((grad*at2)*(grad*lrhoHat))
                                 +0.5*(D_/rhoB_)*((grad*lrho1_)*(grad*c1_))
                                   *atau*(D_/rhoB_)*((grad*at2)*(grad*lrhoHat))
                                 -0.5*k1*dHlrho1_*c1_/(k2_+c1_)*lc
                                   *atau*(D_/rhoB_)*((grad*at2)*(grad*lrhoHat))

                                 -D_*(grad*lc)*(grad*lcHat)
                                 -Hlrho*k1_*k2_/((k2_+c0_)*(k2_+c0_))*lc*lcHat
                                 +(D_/rhoB_)*lrho*((grad*lcHat)*(grad*rho0_))*del,q2)  ;
Expr adjoint2 = adjoint1
          + Integral(interior_, 0.5*lrhoHat*(rho1_ - rhoTarget1_ )*boolExpr_*deltaT, q2)
          + Integral(interior_, 0.5*lrhoHat*(rho0_ - rhoTarget0_ )*boolExpr_*deltaT, q2) ;
```

Biofilm Inversion Formulation and implementation: Approach 2

The PDE constrained optimization problem

$$\min_{\rho,\rho_o} \mathcal{F}(\rho,\rho_u) = \frac{1}{2}\sum_{i=0}^{N}\int_0^T\int_\Omega(\rho-\rho^*)^2\delta(x-x_i)\,\mathbf{dx}\,\mathbf{dt} + \frac{\beta}{\mathbf{2}}\int_\Omega \rho_0^2\,\mathbf{dx}$$

subject to:

$$\begin{aligned}
\frac{\partial\rho}{\partial t} + \frac{D}{\rho_b}\nabla c\cdot\nabla\rho &= 0 && \text{in } \Omega\times[0,T]\\
\rho &= \rho_0 && \text{in } \Omega\times\{t=0\}.\\
D\Delta c &= \frac{k_1 c\rho}{k_2+c} && \text{in } \Omega\\
c &= c_D && \text{in } \Gamma_D
\end{aligned}$$

Additional complications:

- SUPG
- Crank-Nicholson

```
Expr state = Integral(interior_, (rho - rho0_)*rhoHat/deltaT
                              +0.5*(D_/rhoB_)*((grad*rho)*(grad*c))*rhoHat
                              +0.5*(D_/rhoB_)*((grad*rho0_)*(grad*c0_))*rhoHat

                              +(rho - rho0_)/deltaT
                                *stau*(D_/rhoB_)*((grad*st2)*(grad*rhoHat))
                              +0.5*(D_/rhoB_)*((grad*rho)*(grad*c))*rhoHat
                                *stau*(D_/rhoB_)*((grad*st2)*(grad*rhoHat))
                              +0.5*(D_/rhoB_)*((grad*rho0_)*(grad*c0_))*rhoHat
                                *stau*(D_/rhoB_)*((grad*st2)*(grad*rhoHat))

                              -D_*(grad*c)*(grad*cHat)
                              -Hrho*k1_*c/(k2_+c) * c, q2);
```

```
Expr lag = Integral(interior, 0.5*pow(u - target, 2.0)*direcDelta + R*alpha*alpha, q2)
  + Supg_CN_Integral;

Expr lagBC = EssentialBC(top, lambda*u=1.0, q2);

Functional L(mesh, lag, lagBC, vecType);

NonlinearOperator<double> stateProb = F.nonlinearVariationalProb(lambda, lambda0,
                                                                  rho, rho0,
                                                                  alpha, alpha0);

NOXSolver solver(noxParams, stateProb);
solver.solve() ;
```

Preliminary Inversion Results

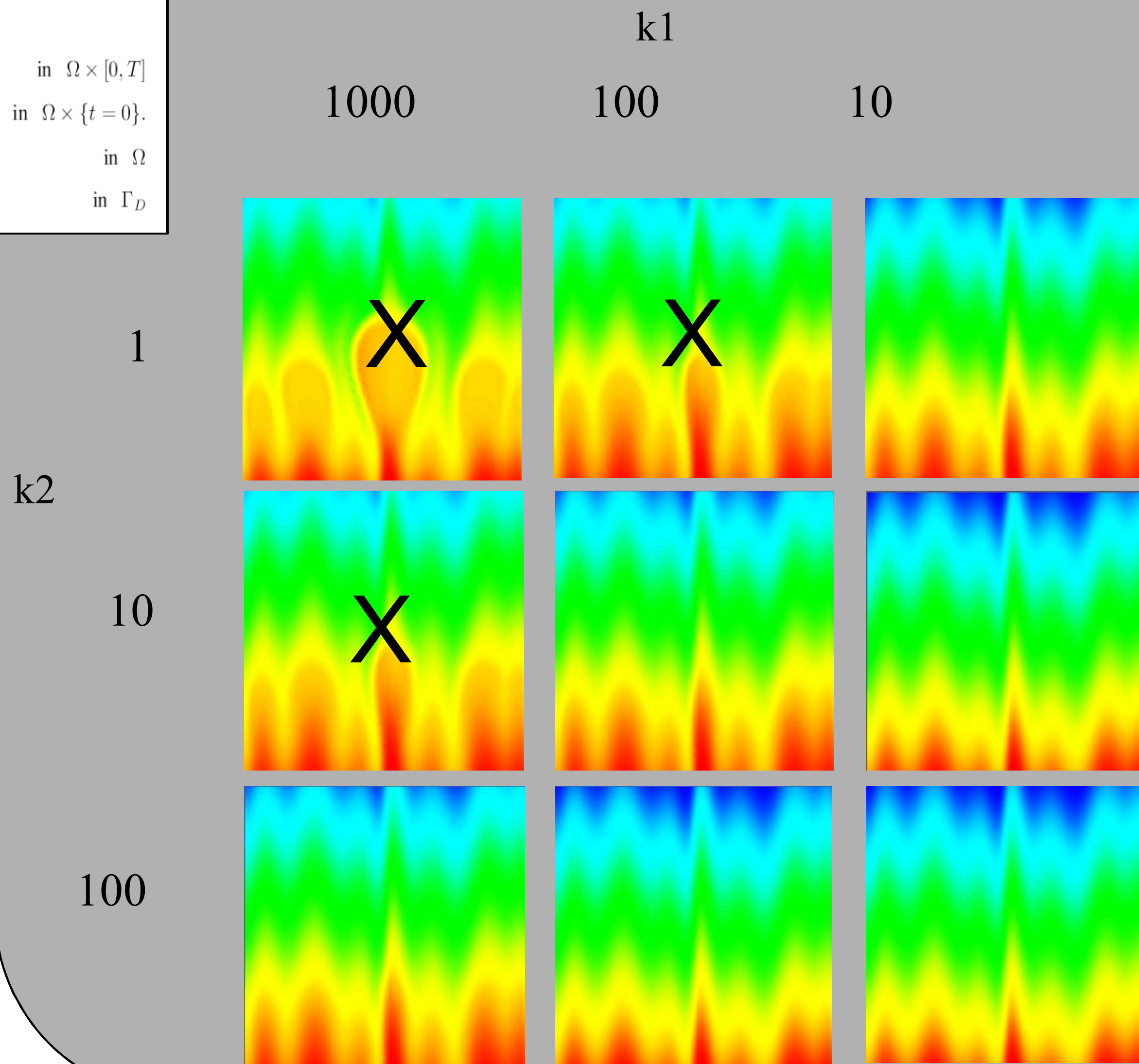
(Realizations shown at final timestep)

The PDE constrained optimization problem

$$\min_{\rho,\rho_o} \mathcal{F}(\rho,\rho_u) = \frac{1}{2}\sum_{i=0}^{N}\int_0^T \int_\Omega (\rho-\rho^*)^2\delta(x-x_i)\,\mathbf{dx}\,\mathbf{dt} + \frac{\beta}{2}\int_\Omega \rho_0^2\,\mathbf{dx}$$

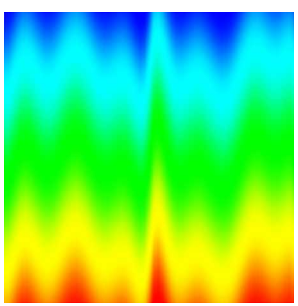
subject to:

$$\begin{aligned}
\frac{\partial \rho}{\partial t} + \frac{D}{\rho_b}\nabla c\cdot\nabla\rho &= 0 && \text{in } \Omega\times[0,T]\\
\rho &= \rho_0 && \text{in } \Omega\times\{t=0\}.\\
D\Delta c &= \frac{k_1 c\rho}{k_2+c} && \text{in } \Omega\\
c &= c_D && \text{in } \Gamma_D
\end{aligned}$$



<u>Inversion Details</u>:

- deltaT = 0.005
- grid 64x64
- R=0.0001
- QN-BFGS



Initial Condition

Sandia National Laboratories

Summary

- Work on biofilm is motivated by range of industrial applications
- Extensive physics defines biofilms dynamics: deposition, transport and detachment
- Inversion of IC or material parameters requires extensive formulation
- Efficient PDECO toolkit isolates users from implementation cost and errors
- Reconstruction of level set function needs to be investigated