DTU Compute
Department of Applied Mathematics and Computer Science

Multi-Scale Simulation and Control in Industrial Biotechnology

Peter Emil Carstensen[1], Teddy Groves[2], Lars Keld Nielsen[2], Ulrich Krühne[3], Krist V. Gernaey[3], John Bagterp Jørgensen[1]

[1]Department of Applied Mathematics and Computer Science, Technical University of Denmark, Denmark
[2]The Novo Nordisk Foundation Center for Biosustainability, Technical University of Denmark, Denmark
[3]Department of Chemical and Biochemical Engineering, Technical University of Denmark, Denmark

1. Introduction

- Aqueous chemical reactions and cellular metabolism are fast relative to the reactor. Both reach equilibrium or quasi-steady state faster than bioreactor processes.

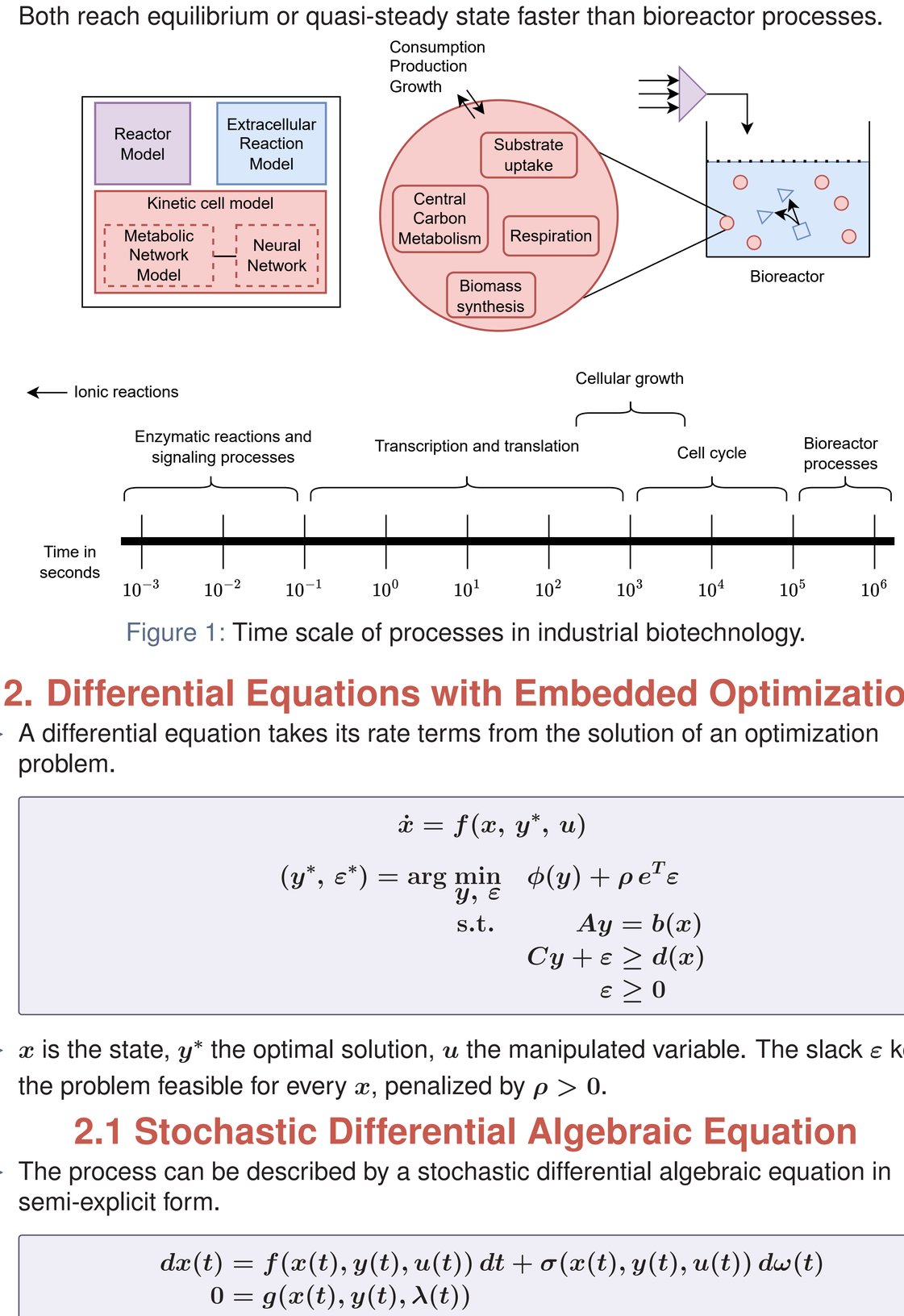

Figure 1: Time scale of processes in industrial biotechnology.

2. Differential Equations with Embedded Optimization

- A differential equation takes its rate terms from the solution of an optimization problem.

$$\dot{x} = f(x,\, y^*,\, u)$$

$$(y^*,\, \varepsilon^*) = \arg\min_{y,\, \varepsilon} \quad \phi(y) + \rho\, e^T \varepsilon$$
$$\text{s.t.} \qquad Ay = b(x)$$
$$Cy + \varepsilon \geq d(x)$$
$$\varepsilon \geq 0$$

- x is the state, y^* the optimal solution, u the manipulated variable. The slack ε keeps the problem feasible for every x, penalized by $\rho > 0$.

2.1 Stochastic Differential Algebraic Equation

- The process can be described by a stochastic differential algebraic equation in semi-explicit form.

$$\begin{aligned} dx(t) &= f(x(t), y(t), u(t))\, dt + \sigma(x(t), y(t), u(t))\, d\omega(t) \\ 0 &= g(x(t), y(t), \lambda(t)) \\ z(t) &= h(x(t), y(t)) \end{aligned}$$

- ω is a Wiener process, σ the diffusion, g the embedded optimality conditions with multipliers λ, and z the measurement.

3. Case A — Chemical Equilibrium, Conductivity, and pH

- At fixed T and P, equilibrium minimizes the Gibbs energy of the mixture. pH follows from the equilibrium composition.

$$\dot{\bar{n}} = \bar{X}_{in} F_{in} - x(\bar{n}) F_{out}$$

$$\min_{n,\, \xi} \quad n^T(\Delta_f g(T, P) + \ln x(n))$$
$$\text{s.t.} \quad \begin{bmatrix} I & -\nu^T \end{bmatrix} \begin{bmatrix} n \\ \xi \end{bmatrix} = B\bar{n}$$
$$n \geq 0$$

- $\bar{n}$ is the state, the reaction-invariant mole numbers. $\bar{X}_{in}$ is the inlet compositions and x the mole fractions.
- n is the equilibrium composition and ξ the extents of reaction. ν is the stoichiometric matrix, and B maps the state into the species vector.

- Consider a set of chemical equilibrium reactions involved in the biochemical production of single cell proteins.

$$\begin{aligned} \mathrm{H_2O} + \mathrm{H_2O} &\rightleftharpoons \mathrm{OH^-} + \mathrm{H_3O^+} \\ \mathrm{H_3PO_4} + \mathrm{H_2O} &\rightleftharpoons \mathrm{H_2PO_4^-} + \mathrm{H_3O^+} \\ \mathrm{H_2PO_4^-} + \mathrm{H_2O} &\rightleftharpoons \mathrm{HPO_4^{2-}} + \mathrm{H_3O^+} \\ \mathrm{HPO_4^{2-}} + \mathrm{H_2O} &\rightleftharpoons \mathrm{PO_4^{3-}} + \mathrm{H_3O^+} \\ \mathrm{CO_2(g)} &\rightleftharpoons \mathrm{CO_2(aq)} \\ \mathrm{CO_2} + \mathrm{H_2O} &\rightleftharpoons \mathrm{H_2CO_3} \\ \mathrm{H_2CO_3} + \mathrm{H_2O} &\rightleftharpoons \mathrm{HCO_3^-} + \mathrm{H_3O^+} \\ \mathrm{HCO_3^-} + \mathrm{H_2O} &\rightleftharpoons \mathrm{CO_3^{2-}} + \mathrm{H_3O^+} \\ \mathrm{NH_3} + \mathrm{H_2O} &\rightleftharpoons \mathrm{OH^-} + \mathrm{NH_4^+} \\ \mathrm{NaOH} &\rightarrow \mathrm{OH^-} + \mathrm{Na^+} \\ \mathrm{HNO_3} + \mathrm{H_2O} &\rightarrow \mathrm{NO_3^-} + \mathrm{H_3O^+} \end{aligned}$$

$$m_i = \frac{n_i}{M_S n_S}, \qquad a_{\mathrm{H_3O^+}} = \frac{m_{\mathrm{H_3O^+}}}{m^0}$$
$$\mathrm{pH} = -\log_{10} a_{\mathrm{H_3O^+}}, \qquad \kappa = \sum_i \lambda_i m_i$$

4. Case B — Dynamic Flux Balance Analysis

- The same structure, with the embedded problem now a parsimonious FBA over the genome-scale network. The cell reaches quasi-steady state much faster than the reactor.

$$\begin{aligned} \dot{V} &= e^T F_{in} - F_{out} \\ \dot{n}_e &= C_{in} F_{in} + R^*_{ext}(c_e) V - c_e F_{out}, & c_e &= \frac{n_e}{V} \\ \dot{n}_X &= R^*_X(c_e) V - c_X F_{out}, & c_X &= \frac{n_X}{V} \\ R^*_{ext}(c_e) &= \nu^T_{ext} r^*(c_e) \\ R^*_X(c_e) &= \nu^T_X r^*(c_e), & r^*(c_e) &= \mu^*(c_e)\, c_X \end{aligned}$$

$$(\mu^*(c_e),\, \xi^*) = \arg \min_{\mu,\, \xi_L,\, \xi_U} \quad e^T\mu + \rho\, e^T(\xi_L + \xi_U)$$
$$\text{s.t.} \quad \begin{bmatrix} \nu_c^T \\ \nu_X^T \end{bmatrix} \mu = \begin{bmatrix} 0 \\ z^*_{FBA}(c_e) \end{bmatrix}$$
$$\mu_{in} \leq q(c_e)$$
$$\begin{bmatrix} I \\ -I \end{bmatrix} \mu + \begin{bmatrix} \xi_L \\ \xi_U \end{bmatrix} \geq \begin{bmatrix} \mu_{min} \\ -\mu_{max} \end{bmatrix}$$
$$\mu,\, \xi_L,\, \xi_U \geq 0$$

- $e^T\mu$ is the total flux. Minimizing it at fixed growth gives the parsimonious solution.
- $q(c_e)$ is the kinetic uptake limit and $z^*_{FBA}(c_e)$ the growth rate from a prior FBA. The slacks keep the LP feasible as substrate depletes.

5. Neural Network Surrogate

- The embedded LP is replaced by direct evaluation of a trained network.

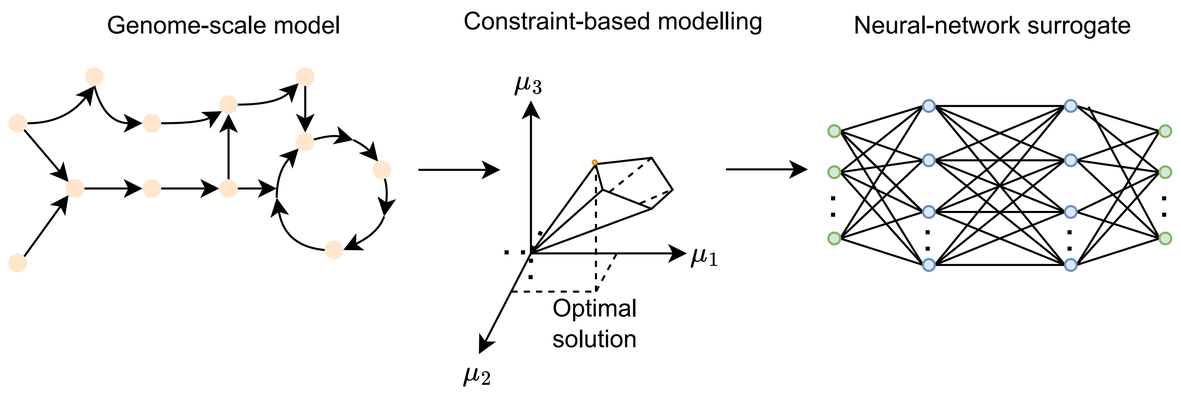

Figure 2: The genome-scale network defines a constraint-based problem whose solution lies on a face of the flux polytope. A neural network is trained on these solutions to approximate the flux map.

$$\mu^*(c_e) = \mathrm{NN}(q(c_e), \theta)$$

- Only the flux map changes. The differential states remains the same, and the surrogate can be directly placed into the dynamic model.
- $\mathrm{NN}(\cdot, \theta)$ is a smooth map, so its Jacobian is continuous and available by automatic differentiation. The LP solution map is only piecewise affine, and its Jacobian has discontinuities at each change of active set.
- The embedded optimization problem is replaced by a single forward pass through the neural network.

6. Conclusion

- Case A needs no surrogate. The Gibbs objective is smooth and strictly convex, so the KKT conditions give an index-1 DAE and the sensitivities follow from the factorization the Newton step already uses.

$$\begin{aligned} g(T, P, n) &= n^T \nabla_n g(T, P, n) \\ \nabla_n g(T, P, n) &= \Delta_f g(T, P) + \ln x(n) \\ [\nabla^2_{nn} g(T, P, n)]_{ij} &= \begin{cases} \frac{1}{n_i} - \frac{1}{e^T n} & i = j \\ -\frac{1}{e^T n} & i \neq j \end{cases} \end{aligned}$$

- Case B is a linear program, which has no Hessian and moves its active set instead. The surrogate replaces only the flux map, and trades the explicit constraints of the LP for a smooth and fast approximation.
- Both models now have a differentiable right-hand side, by KKT sensitivities for the equilibrium problem and by automatic differentiation for the surrogate. Each is ready for state estimation and predictive control

Contacts: Peter E. Carstensen: pecar@dtu.dk, John B. Jørgensen: jbjo@dtu.dk