Product Dynamic Transitions Using A Derivative-Free Optimization Approach

Antonio Flores-Tlacuahuac

Department of Chemical Engineering

Universidad Iberoamericana

México

with colaborations from:

Israel Negrellos-Ortiz

January 4, 2016

Outline

- Introduction

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions
- DFO Approaches for Product Transitions

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions
- DFO Approaches for Product Transitions
  - Unconstrained optimization problem with penalized model

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions
- DFO Approaches for Product Transitions
  - Unconstrained optimization problem with penalized model
  - Model Predictive control using black box dynamic modeling

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions
- DFO Approaches for Product Transitions
  - Unconstrained optimization problem with penalized model
  - Model Predictive control using black box dynamic modeling
- Problem definition

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions
- DFO Approaches for Product Transitions
  - Unconstrained optimization problem with penalized model
  - Model Predictive control using black box dynamic modeling
- Problem definition
- Examples

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions
- DFO Approaches for Product Transitions
  - Unconstrained optimization problem with penalized model
  - Model Predictive control using black box dynamic modeling
- Problem definition
- Examples
  - Fundamental differential equation

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions
- DFO Approaches for Product Transitions
  - Unconstrained optimization problem with penalized model
  - Model Predictive control using black box dynamic modeling
- Problem definition
- Examples
  - Fundamental differential equation
  - Hicks reactor

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions
- DFO Approaches for Product Transitions
  - Unconstrained optimization problem with penalized model
  - Model Predictive control using black box dynamic modeling
- Problem definition
- Examples
  - Fundamental differential equation
  - Hicks reactor
  - Binary distillation column in the ASPEN simulator

Outline

- Introduction
- Traditional Optimal Model-based dynamic transitions
- DFO Approaches for Product Transitions
  - Unconstrained optimization problem with penalized model
  - Model Predictive control using black box dynamic modeling
- Problem definition
- Examples
  - Fundamental differential equation
  - Hicks reactor
  - Binary distillation column in the ASPEN simulator
- Conclusions

Introduction

Introduction

- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.

Introduction

- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.
- The problem consists in computing the time domain values of the control variables $\mathbf{u}(t)$ such that the system response $\mathbf{y}(t)$ attains a desired value embedded in an objective function $\Omega(\mathbf{y}, \mathbf{u})$.

Introduction

- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.
- The problem consists in computing the time domain values of the control variables $\mathbf{u}(t)$ such that the system response $\mathbf{y}(t)$ attains a desired value embedded in an objective function $\Omega(\mathbf{y}, \mathbf{u})$.
- Optimal product transitions can be calculated systematically by setting the product transition problem as a dynamic optimization problem.

Introduction

- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.
- The problem consists in computing the time domain values of the control variables $\mathbf{u}(t)$ such that the system response $\mathbf{y}(t)$ attains a desired value embedded in an objective function $\Omega(\mathbf{y}, \mathbf{u})$.
- Optimal product transitions can be calculated systematically by setting the product transition problem as a dynamic optimization problem.
- Most of these optimization algorithm assume that the gradient of the objective function and related constraints are someway available.

Introduction

- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.
- The problem consists in computing the time domain values of the control variables $\mathbf{u}(t)$ such that the system response $\mathbf{y}(t)$ attains a desired value embedded in an objective function $\Omega(\mathbf{y}, \mathbf{u})$.
- Optimal product transitions can be calculated systematically by setting the product transition problem as a dynamic optimization problem.
- Most of these optimization algorithm assume that the gradient of the objective function and related constraints are someway available.
- There are some practical optimization problems where derivatives are not available or are not reliable making impossible or undesirable to use gradient-based optimization techniques.

Introduction

- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.
- The problem consists in computing the time domain values of the control variables $\mathbf{u}(t)$ such that the system response $\mathbf{y}(t)$ attains a desired value embedded in an objective function $\Omega(\mathbf{y}, \mathbf{u})$.
- Optimal product transitions can be calculated systematically by setting the product transition problem as a dynamic optimization problem.
- Most of these optimization algorithm assume that the gradient of the objective function and related constraints are someway available.
- There are some practical optimization problems where derivatives are not available or are not reliable making impossible or undesirable to use gradient-based optimization techniques.
- Such situations has to do with either real industrial or lab-scale processes or with normally huge models embedded in simulation platforms.

Introduction

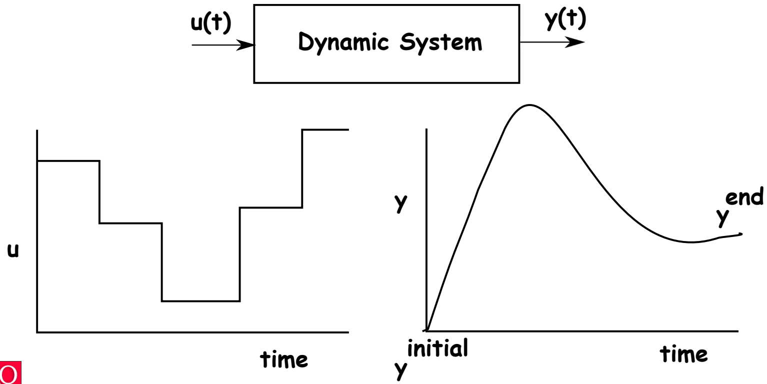
- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.
- The problem consists in computing the time domain values of the control variables $\mathbf{u}(t)$ such that the system response $\mathbf{y}(t)$ attains a desired value embedded in an objective function $\Omega(\mathbf{y}, \mathbf{u})$.
- Optimal product transitions can be calculated systematically by setting the product transition problem as a dynamic optimization problem.
- Most of these optimization algorithm assume that the gradient of the objective function and related constraints are someway available.
- There are some practical optimization problems where derivatives are not available or are not reliable making impossible or undesirable to use gradient-based optimization techniques.
- Such situations has to do with either real industrial or lab-scale processes or with normally huge models embedded in simulation platforms.
- We deal with extensions of trust-region methods, which are a subset of deterministic DFO algorithms, to address the solution optimization problems where no model gradient information is available.

Introduction

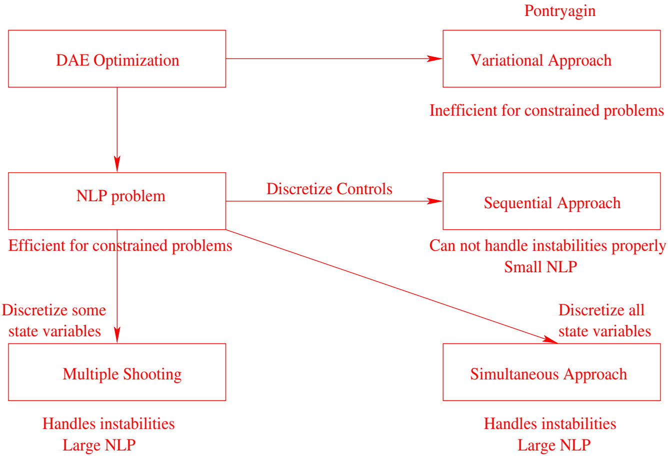
- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.
- The problem consists in computing the time domain values of the control variables $\mathbf{u}(t)$ such that the system response $\mathbf{y}(t)$ attains a desired value embedded in an objective function $\Omega(\mathbf{y}, \mathbf{u})$.
- Optimal product transitions can be calculated systematically by setting the product transition problem as a dynamic optimization problem.
- Most of these optimization algorithm assume that the gradient of the objective function and related constraints are someway available.
- There are some practical optimization problems where derivatives are not available or are not reliable making impossible or undesirable to use gradient-based optimization techniques.
- Such situations has to do with either real industrial or lab-scale processes or with normally huge models embedded in simulation platforms.
- We deal with extensions of trust-region methods, which are a subset of deterministic DFO algorithms, to address the solution optimization problems where no model gradient information is available.

Traditional Optimal Model-based dynamic transitions

Take a dynamic system from an initial point to a final point in the best possible way



Common Approaches for Solving Dynamic Optimization Problems



Discretizing ODEs using Orthogonal Collocation

Given an ODE system:

$$\frac{dx}{dt} = f(x, u, p), \quad x(0) = x_{init}$$

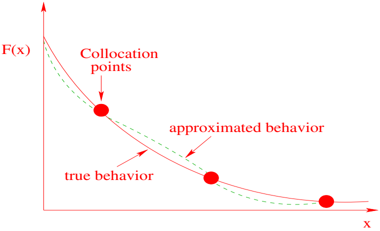
where $x(t)$ are the system states, $u(t)$ is the manipulated variable and p are the system parameters.

The aim is to approximate the behaviour of x and u by Lagrange interpolation polynomials (of orders $\mathcal{K}+1$ and $\mathcal{K}$, respectively) at collocation or discretization points t_k:

$$x_{k+1}(t) = \sum_{k=0}^{\mathcal{K}} x_k \ell_k^x(t), \quad \ell_k^x(t) = \prod_{\substack{j=0 \\ j\neq k}}^{\mathcal{K}} \frac{t - t_j}{t_k - t_j}$$

$$u_k(t) = \sum_{k=1}^{\mathcal{K}} u_k \ell_k^u(t), \quad \ell_k^u(t) = \prod_{\substack{j=1 \\ j\neq k}}^{\mathcal{K}} \frac{t - t_j}{t_k - t_j}$$

$$x_{N+1}(t_k) = x_k, \quad u_N(t_k) = u_k$$



Therefore replacing into the original ODE system, we get the system residual $\mathcal{R}(t_k)$:

$$\mathcal{R}(t_k) = \sum_{j=0}^{\mathcal{K}} x_j \frac{d\ell_j(t_k)}{dt} - f(x_k, u_k, p) = 0, \quad k = 1, .., \mathcal{K}$$

Transformation of a Dynamic Optimization problem into a NLP

Original dynamic optimization problem

$$\min_{x,u} \phi(x, u)$$

s.t.

$$\frac{dx(t)}{dt} = F\left(x(t), u(t), t, p\right)$$
$$x(0) = x^0$$
$$g(x(t), u(t), p) \leq 0$$
$$h(x(t), u(t), p) = 0$$
$$x^l \leq x \leq x^u$$
$$u^l \leq u \leq u^u$$

Discretized NLP

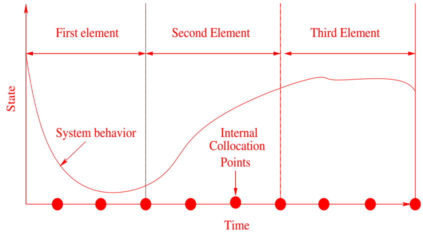
$$\min_{x_k, u_k} \phi(x_k, u_k)$$

s.t.

$$\sum_{j=0}^{\mathcal{K}} x_j \dot{\ell}_j(t_k) - F\left(x_k, u_k\right) = 0, \;\; k = 1, \ldots, \mathcal{K}$$
$$x_0 = x(0)$$
$$g(x_k, u_k, p) \leq 0, \;\; k = 1, \ldots, \mathcal{K}$$
$$h(x_k, u_k, p) = 0, \;\; k = 1, \ldots, \mathcal{K}$$
$$x^l \leq x \leq x^u$$
$$u^l \leq u \leq u^u$$

Approximation of a Dynamic Optimization Problem using Orthogonal Collocation on Finite Elements

Sometimes it is convenient to use Orthogonal Collocation on Finite Elements to approximate the behavior of systems exhibiting fast dynamics.



$$\min_{x_k, u_k} \phi(x, u)$$

s.t.

$$\sum_{j=0}^{\mathcal{K}} x_{ij} \dot{\ell}_j(\tau_k) - h_i F(x_{ik}, u_{ik}) = 0, \quad k \overset{i=1,\ldots,NE}{=} 1, \ldots, NC$$

$$x_{10} = x(0)$$

$$g(x_{ik}, u_{ik}, p) = 0, \quad i = 1, \ldots, NE; k = 1, .., NC$$

$$x_{ij}^{l} \leq x_{ij} \leq x_{ij}^{u}, \quad i = 1, \ldots, NE; k = 1, .., NC$$

$$u_{ij}^{l} \leq u_{ij} \leq u_{ij}^{u}, \quad i = 1, \ldots, NE; k = 1, .., NC$$

where NE is the number of finite elements, NC is the number of internal collocation points, h_i is the length of each element.

Example: Dynamic optimal transition between two steady-states: Hicks CSTR

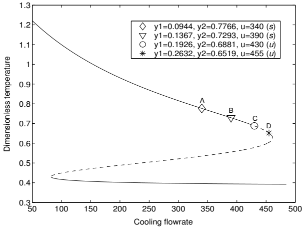
$$\frac{dC}{dt} = \frac{1-C}{\theta} - k_{10}e^{-N/T}C$$

$$\frac{dT}{dt} = \frac{y_f - T}{\theta} + k_{10}e^{-N/T}C - \alpha U(T - y_c)$$

Parameter values

θ	20	Residence time
T_f	300	Feed temperature
J	100	$(-\Delta H)/(\rho C_p)$
k_{10}	300	Preexponential factor
c_f	7.6	Feed concentration
T_c	290	Coolant temperature
α	1.95×10^{-4}	Heat transfer area
N	5	$E_1/(RJc_f)$

Desired Transition B $\rightarrow$ A



Desired dynamic transition

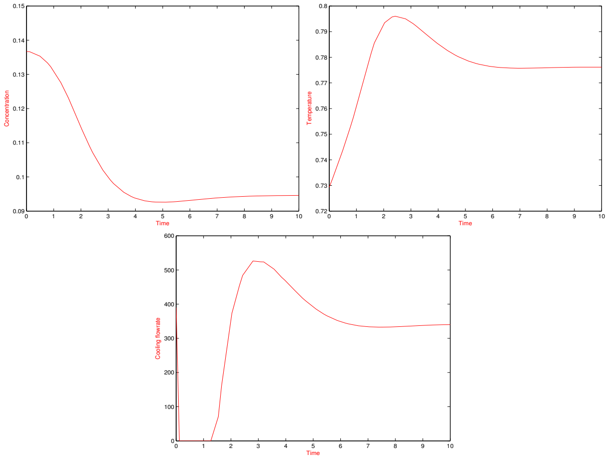
	C	T	U
Initial (B)	0.1367	0.7293	390
Final (A)	0.0944	0.7766	340

C = Concentration (c/c_f), T = temperature (T_r/Jc_f), y_c = Coolant temperature (T_c/Jc_f), y_f = feed temperature (T_f/Jc_f), U = Cooling flowrate. c and T_r are nondimensionless concentration and reactor temperature.

Dynamic Transitions profiles for the Hicks CSTR example

As objective function the requirement of minimum transition time between the initial and final steady-states will be imposed:

$$\text{Min} \int_0^{t_f} \left\{ \alpha_1 (C(t) - C_{des})^2 + \alpha_2 (T(t) - T_{des})^2 + \alpha_3 (U(t) - U_{des})^2 \right\} dt$$



DFO Approaches for Product Transitions

DFO Approaches for Product Transitions

- Unconstrained optimization problem with penalized model.

DFO Approaches for Product Transitions

- Unconstrained optimization problem with penalized model.
  - Product transition problems were solved using a dynamic model to compare NLP and DFO solutions.

DFO Approaches for Product Transitions

- Unconstrained optimization problem with penalized model.
  - Product transition problems were solved using a dynamic model to compare NLP and DFO solutions.
  - The constrained dynamic optimization problem was transformed into an unconstrained optimization problem,

DFO Approaches for Product Transitions

- Unconstrained optimization problem with penalized model.
  - Product transition problems were solved using a dynamic model to compare NLP and DFO solutions.
  - The constrained dynamic optimization problem was transformed into an unconstrained optimization problem,
  - The set of ODE constraints related were approximated using the transcription approach.

DFO Approaches for Product Transitions

- Unconstrained optimization problem with penalized model.
  - Product transition problems were solved using a dynamic model to compare NLP and DFO solutions.
  - The constrained dynamic optimization problem was transformed into an unconstrained optimization problem,
  - The set of ODE constraints related were approximated using the transcription approach.
  - Optimization problems solved using nonlinear optimization codes and the BOBYQA DFO solver.

DFO Approaches for Product Transitions

- Unconstrained optimization problem with penalized model.
  - Product transition problems were solved using a dynamic model to compare NLP and DFO solutions.
  - The constrained dynamic optimization problem was transformed into an unconstrained optimization problem,
  - The set of ODE constraints related were approximated using the transcription approach.
  - Optimization problems solved using nonlinear optimization codes and the BOBYQA DFO solver.
  - Product transitions are open-loop or off-line control policies.

DFO Approaches for Product Transitions

- Unconstrained optimization problem with penalized model.
  - Product transition problems were solved using a dynamic model to compare NLP and DFO solutions.
  - The constrained dynamic optimization problem was transformed into an unconstrained optimization problem,
  - The set of ODE constraints related were approximated using the transcription approach.
  - Optimization problems solved using nonlinear optimization codes and the BOBYQA DFO solver.
  - Product transitions are open-loop or off-line control policies.

- Model Predictive control using black box dynamic modeling

- Model Predictive control using black box dynamic modeling
  - Optimal closed-loop transitions can be efficiently assessed using Model Predictive Control (MPC) techniques.

- Model Predictive control using black box dynamic modeling
  - Optimal closed-loop transitions can be efficiently assessed using Model Predictive Control (MPC) techniques.
  - MPC is a closed-loop control technique well suited for the control of nonlinear systems using a receding horizon approach.

- Model Predictive control using black box dynamic modeling
  - Optimal closed-loop transitions can be efficiently assessed using Model Predictive Control (MPC) techniques.
  - MPC is a closed-loop control technique well suited for the control of nonlinear systems using a receding horizon approach.
  - A set of values of the input or manipulated variables (i.e. $\mathbf{u_1}, \ldots, \mathbf{u_N}$) are computed such that the output or controlled variables (i.e. $\mathbf{x_1}, \ldots, \mathbf{x_M}$) are forced towards target or desired values.

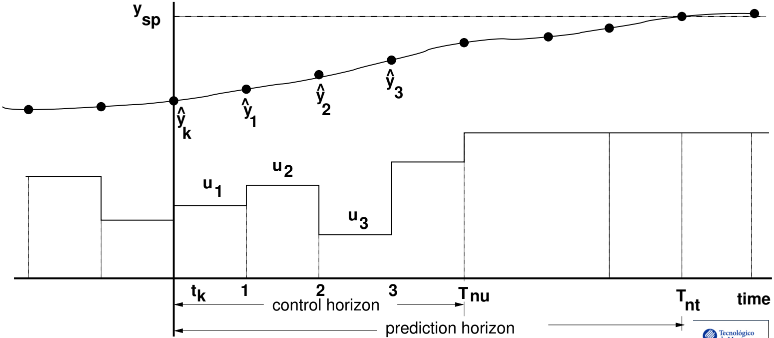
- Model Predictive control using black box dynamic modeling
  - Optimal closed-loop transitions can be efficiently assessed using Model Predictive Control (MPC) techniques.
  - MPC is a closed-loop control technique well suited for the control of nonlinear systems using a receding horizon approach.
  - A set of values of the input or manipulated variables (i.e. $\mathbf{u_1}, \ldots, \mathbf{u_N}$) are computed such that the output or controlled variables (i.e. $\mathbf{x_1}, \ldots, \mathbf{x_M}$) are forced towards target or desired values.
  - Commonly, only the first control action $\mathbf{u_1}$ is applied to the controlled system.

Model Predictive control using black box dynamic modeling

- Optimal closed-loop transitions can be efficiently assessed using Model Predictive Control (MPC) techniques.
- MPC is a closed-loop control technique well suited for the control of nonlinear systems using a receding horizon approach.
- A set of values of the input or manipulated variables (i.e. $\mathbf{u_1}, \ldots, \mathbf{u_N}$) are computed such that the output or controlled variables (i.e. $\mathbf{x_1}, \ldots, \mathbf{x_M}$) are forced towards target or desired values.
- Commonly, only the first control action $\mathbf{u_1}$ is applied to the controlled system.
- When a new measurement becomes available, the control scenario is moved ahead by a time equivalent to the sampling time and the procedure is repeated until the target values are obtained.

Model Predictive control using black box dynamic modeling

- Optimal closed-loop transitions can be efficiently assessed using Model Predictive Control (MPC) techniques.
- MPC is a closed-loop control technique well suited for the control of nonlinear systems using a receding horizon approach.
- A set of values of the input or manipulated variables (i.e. $\mathbf{u_1}, \ldots, \mathbf{u_N}$) are computed such that the output or controlled variables (i.e. $\mathbf{x_1}, \ldots, \mathbf{x_M}$) are forced towards target or desired values.
- Commonly, only the first control action $\mathbf{u_1}$ is applied to the controlled system.
- When a new measurement becomes available, the control scenario is moved ahead by a time equivalent to the sampling time and the procedure is repeated until the target values are obtained.



- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.

- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.
- We deploy a kind of DFO algorithms based on using trust-region optimization techniques.

- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.
- We deploy a kind of DFO algorithms based on using trust-region optimization techniques.
- In the case of DFO trust-region methods the objective function is approximated by a quadratic function.

- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.
- We deploy a kind of DFO algorithms based on using trust-region optimization techniques.
- In the case of DFO trust-region methods the objective function is approximated by a quadratic function.
- Such an approximation of the objective function is built using only input and output information .

- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.
- We deploy a kind of DFO algorithms based on using trust-region optimization techniques.
- In the case of DFO trust-region methods the objective function is approximated by a quadratic function.
- Such an approximation of the objective function is built using only input and output information .
- Using DFO techniques no information, in the form of an explicit dynamic model or their related derivatives, is required for the dynamic optimization of processing systems.

- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.
- We deploy a kind of DFO algorithms based on using trust-region optimization techniques.
- In the case of DFO trust-region methods the objective function is approximated by a quadratic function.
- Such an approximation of the objective function is built using only input and output information .
- Using DFO techniques no information, in the form of an explicit dynamic model or their related derivatives, is required for the dynamic optimization of processing systems.
- We only require (a) input/output information of the controlled system and (b) solving the dynamic optimization problem by approximating the objective function using DFO trust-region optimization techniques.

- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.
- We deploy a kind of DFO algorithms based on using trust-region optimization techniques.
- In the case of DFO trust-region methods the objective function is approximated by a quadratic function.
- Such an approximation of the objective function is built using only input and output information .
- Using DFO techniques no information, in the form of an explicit dynamic model or their related derivatives, is required for the dynamic optimization of processing systems.
- We only require (a) input/output information of the controlled system and (b) solving the dynamic optimization problem by approximating the objective function using DFO trust-region optimization techniques.
- The issue consists in formulating an objective function which mimics the model predictive control approach used when a dynamic model is available.

- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.
- We deploy a kind of DFO algorithms based on using trust-region optimization techniques.
- In the case of DFO trust-region methods the objective function is approximated by a quadratic function.
- Such an approximation of the objective function is built using only input and output information .
- Using DFO techniques no information, in the form of an explicit dynamic model or their related derivatives, is required for the dynamic optimization of processing systems.
- We only require (a) input/output information of the controlled system and (b) solving the dynamic optimization problem by approximating the objective function using DFO trust-region optimization techniques.
- The issue consists in formulating an objective function which mimics the model predictive control approach used when a dynamic model is available.

$$\Omega = \sum_{i=1}^{N}[(x_i - x^f)^2 + (u_i - u^f)^2] + \lambda_x \sum_{i=1}^{N}[\Delta x_i]^2 + \lambda_u \sum_{i=1}^{N}[\Delta u_i]^2$$

where the superscript f stands for final, desired or target values, λ_x, λ_u are penalties impossed on variations of the magnitude of the states and the manipulated variables, respectively.

- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.
- We deploy a kind of DFO algorithms based on using trust-region optimization techniques.
- In the case of DFO trust-region methods the objective function is approximated by a quadratic function.
- Such an approximation of the objective function is built using only input and output information .
- Using DFO techniques no information, in the form of an explicit dynamic model or their related derivatives, is required for the dynamic optimization of processing systems.
- We only require (a) input/output information of the controlled system and (b) solving the dynamic optimization problem by approximating the objective function using DFO trust-region optimization techniques.
- The issue consists in formulating an objective function which mimics the model predictive control approach used when a dynamic model is available.

$$\Omega = \sum_{i=1}^{N} [(x_i - x^f)^2 + (u_i - u^f)^2] + \lambda_x \sum_{i=1}^{N} [\Delta x_i]^2 + \lambda_u \sum_{i=1}^{N} [\Delta u_i]^2$$

  where the superscript f stands for final, desired or target values, λ_x, λ_u are penalties impossed on variations of the magnitude of the states and the manipulated variables, respectively.
- The aim of the last two terms is to penalize for large changes in both the states and manipulated variable values.

- DFO algorithms can be used when a dynamic model or derivatives of the underlying systems are not available.
- We deploy a kind of DFO algorithms based on using trust-region optimization techniques.
- In the case of DFO trust-region methods the objective function is approximated by a quadratic function.
- Such an approximation of the objective function is built using only input and output information .
- Using DFO techniques no information, in the form of an explicit dynamic model or their related derivatives, is required for the dynamic optimization of processing systems.
- We only require (a) input/output information of the controlled system and (b) solving the dynamic optimization problem by approximating the objective function using DFO trust-region optimization techniques.
- The issue consists in formulating an objective function which mimics the model predictive control approach used when a dynamic model is available.

$$\Omega = \sum_{i=1}^{N} [(x_i - x^f)^2 + (u_i - u^f)^2] + \lambda_x \sum_{i=1}^{N} [\Delta x_i]^2 + \lambda_u \sum_{i=1}^{N} [\Delta u_i]^2$$

  where the superscript f stands for final, desired or target values, λ_x, λ_u are penalties impossed on variations of the magnitude of the states and the manipulated variables, respectively.
- The aim of the last two terms is to penalize for large changes in both the states and manipulated variable values.

$$\Delta x_i = x_i - x_{i-1}$$
$$\Delta u_i = u_i - u_{i-1}$$

In summary, the computation of optimal product dynamic transitions using a DFO approach can be cast as follows:

$$\min_{\mathbf{u}} \Omega = \sum_{i=1}^{N}[(x_i - x^f)^2 + (u_i - u^f)^2] + \lambda_x \sum_{i=1}^{N}[\Delta x_i]^2 + \lambda_u \sum_{i=1}^{N}[\Delta u_i]^2$$

$$\begin{aligned}
\text{s.t.} \qquad & x^l \le x_i \le x^u, & i = 1, \ldots, N \\
& u^l \le u_i \le u^u, & i = 1, \ldots, N \\
& \Delta x^l \le \Delta x_i \le \Delta x^u, & i = 1, \ldots, N \\
& \Delta u^l \le \Delta u_i \le \Delta u^u, & i = 1, \ldots, N
\end{aligned}$$

where the l and u superscripts stand for lower and upper bounds, respectively.

Problem definition

Problem definition

Given:

- An objective function representing the target grade or product plant transitions.

Problem definition

Given:

- An objective function representing the target grade or product plant transitions.
- A black box system (either industrial or laboratory system or computer simulation) from which only input and output information about the dynamic performance of such a system is recorded.

Problem definition

Given:

- An objective function representing the target grade or product plant transitions.
- A black box system (either industrial or laboratory system or computer simulation) from which only input and output information about the dynamic performance of such a system is recorded.
- A set of simple box bound constraints on the system states.

Problem definition

Given:

- An objective function representing the target grade or product plant transitions.
- A black box system (either industrial or laboratory system or computer simulation) from which only input and output information about the dynamic performance of such a system is recorded.
- A set of simple box bound constraints on the system states.
- We assume that numerical values of the system derivatives of the objective function are not available.

Problem definition

Given:

- An objective function representing the target grade or product plant transitions.
- A black box system (either industrial or laboratory system or computer simulation) from which only input and output information about the dynamic performance of such a system is recorded.
- A set of simple box bound constraints on the system states.
- We assume that numerical values of the system derivatives of the objective function are not available.

Example 1: Fundamental differential equation

Example 1: Fundamental differential equation

$$\frac{dy}{dt} = -ay(t) + bu(t), \;\; a = b = 1$$

where y is the state variable, t is the independent variable, u is a control variable.

Example 1: Fundamental differential equation

$$\frac{dy}{dt} = -ay(t) + bu(t), \; a = b = 1$$

where y is the state variable, t is the independent variable, u is a control variable.

- Penalized state transition objective function, which includes the discretization of the dynamic mathematical model, reads:

$$\min \, \Omega = \frac{1}{2}\int_0^1 (y^2 + u^2)dt + \mathbf{F}(y, \dot{y}, \lambda)$$

where $\mathbf{F}(y, \dot{y}, \lambda)$ stands for a function which represents the discretized dynamic model as a function of the states (y), states derivates $(\dot{x})$ and penalty parameters (λ).

Example 1: Fundamental differential equation

$$\frac{dy}{dt} = -ay(t) + bu(t), \ a = b = 1$$

where y is the state variable, t is the independent variable, u is a control variable.

- Penalized state transition objective function, which includes the discretization of the dynamic mathematical model, reads:

$$\min \ \Omega = \frac{1}{2}\int_0^1 (y^2 + u^2)dt + \mathbf{F}(y, \dot{y}, \lambda)$$

  where $\mathbf{F}(y, \dot{y}, \lambda)$ stands for a function which represents the discretized dynamic model as a function of the states (y), states derivates $(\dot{x})$ and penalty parameters (λ).

- Transitions as computed by the CONOPT NLP and the BOBYQA DFO solvers

Example 1: Fundamental differential equation

$$\frac{dy}{dt} = -ay(t) + bu(t), \;\; a = b = 1$$

where y is the state variable, t is the independent variable, u is a control variable.

- Penalized state transition objective function, which includes the discretization of the dynamic mathematical model, reads:

  $$\min \; \Omega = \frac{1}{2}\int_0^1 (y^2 + u^2)dt + \mathbf{F}(y, \dot{y}, \lambda)$$

  where $\mathbf{F}(y, \dot{y}, \lambda)$ stands for a function which represents the discretized dynamic model as a function of the states (y), states derivates ($\dot{x}$) and penalty parameters (λ).
- Transitions as computed by the CONOPT NLP and the BOBYQA DFO solvers
- The vectors of initial $[y_0, u_0]$ and target $[y^f, u^f]$ transition conditions are given by $[-0.375, 1]$ and $[3, 0]$, respectively.

Example 1: Fundamental differential equation

$$\frac{dy}{dt} = -ay(t) + bu(t), \;\; a = b = 1$$

where y is the state variable, t is the independent variable, u is a control variable.

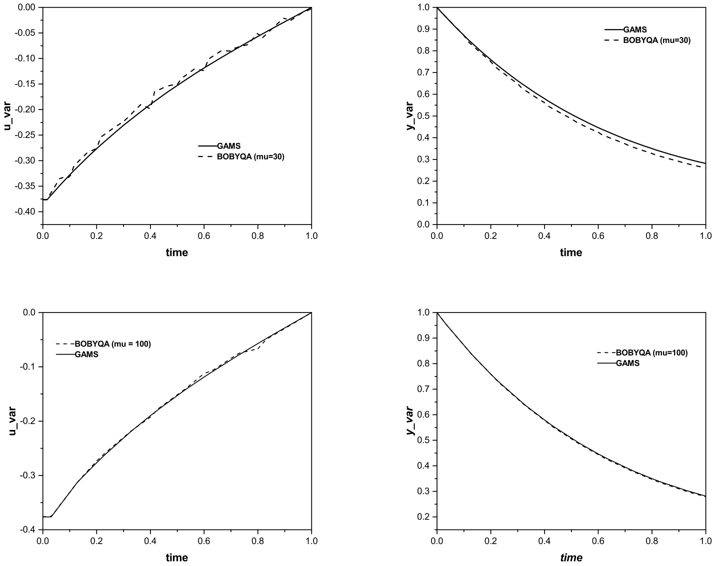
- Penalized state transition objective function, which includes the discretization of the dynamic mathematical model, reads:

$$\min \; \Omega = \frac{1}{2}\int_0^1 (y^2 + u^2)dt + \mathbf{F}(y, \dot{y}, \lambda)$$

  where $\mathbf{F}(y, \dot{y}, \lambda)$ stands for a function which represents the discretized dynamic model as a function of the states (y), states derivates ($\dot{x}$) and penalty parameters (λ).

- Transitions as computed by the CONOPT NLP and the BOBYQA DFO solvers

- The vectors of initial $[y_0, u_0]$ and target $[y^f, u^f]$ transition conditions are given by $[-0.375, 1]$ and $[3, 0]$, respectively.



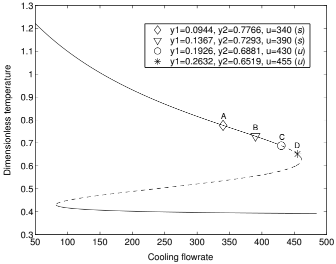
Upper: 10 finite elements and $\mu = 30$, Lower: 5 finite elements and $\mu = 100$.

Example 2: Hicks reactor

Example 2: Hicks reactor

$$\frac{dy_1}{dt} = \frac{1 - y_1}{\theta} - k_{10} e^{-N/y_2} y_1$$
$$\frac{dy_2}{dt} = \frac{y_f - y_2}{\theta} - k_{10} e^{-N/y_2} y_1 - \alpha U (y_2 - y_c)$$

Parameter	Value	Description
θ	20	Residence time
J	100	$(-\Delta H)/(\rho C_p)$
c_f	27.6	Feed concentration
α	$1.95e^{-4}$	Dimensionless heat transfer area
T_f	300	Feed temperature
k_{10}	300	Preexponential factor
T_c	290	Coolant temperature
N	5	$E_1/(RJc_f)$



- Objective function

- Objective function

$$\min \Omega = \int_0^t \{(y_1(t) - y_1^d)^2 + (y_2(t) - y_2^d)^2 + (u(t) - u^d)^2\} dt + \mathbf{F}(\mathbf{y}, \dot{\mathbf{y}}, \lambda)$$

where y_1 is the dimensionless concentration, y_2 is the dimensionless temperature and u is the cooling water flowrate, the superscript d stands for desired or target values. The term $\mathbf{F}(\mathbf{y}, \dot{\mathbf{y}}, \lambda)$ is a function representing the discretized dynamic model.

- Objective function

$$\min\ \Omega = \int_0^t \{(y_1(t) - y_1^d)^2 + (y_2(t) - y_2^d)^2 + (u(t) - u^d)^2\} dt + \mathbf{F}(\mathbf{y}, \dot{\mathbf{y}}, \lambda)$$

  where y_1 is the dimensionless concentration, y_2 is the dimensionless temperature and u is the cooling water flowrate, the superscript d stands for desired or target values. The term $\mathbf{F}(\mathbf{y}, \dot{\mathbf{y}}, \lambda)$ is a function representing the discretized dynamic model.
- A product transition from steady-state B to the steady-state A will be carried out.

- Objective function

$$\min\, \Omega = \int_0^t \{(y_1(t) - y_1^d)^2 + (y_2(t) - y_2^d)^2 + (u(t) - u^d)^2\} dt + \mathbf{F}(\mathbf{y}, \dot{\mathbf{y}}, \lambda)$$

where y_1 is the dimensionless concentration, y_2 is the dimensionless temperature and u is the cooling water flowrate, the superscript d stands for desired or target values. The term $\mathbf{F}(\mathbf{y}, \dot{\mathbf{y}}, \lambda)$ is a function representing the discretized dynamic model.

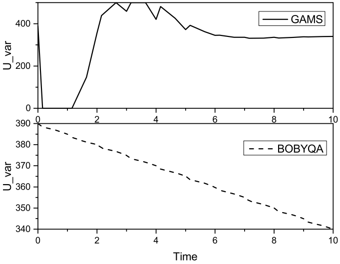
- A product transition from steady-state B to the steady-state A will be carried out.
- 10 finite elements with three internal collocation points were used for the discretization of the dynamic model.

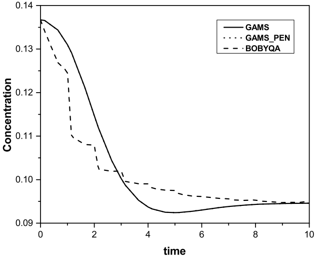
- Objective function

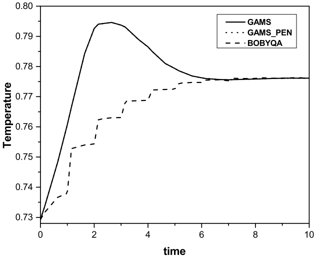
$$\min\ \Omega = \int_0^t \{(y_1(t)-y_1^d)^2+(y_2(t)-y_2^d)^2+(u(t)-u^d)^2\}dt+\mathbf{F}(\mathbf{y},\dot{\mathbf{y}},\lambda)$$

where y_1 is the dimensionless concentration, y_2 is the dimensionless temperature and u is the cooling water flowrate, the superscript d stands for desired or target values. The term $\mathbf{F}(\mathbf{y},\dot{\mathbf{y}},\lambda)$ is a function representing the discretized dynamic model.

- A product transition from steady-state B to the steady-state A will be carried out.
- 10 finite elements with three internal collocation points were used for the discretization of the dynamic model.


GAMS
BOBYQA
U_var
U_var
0
200
400
340
350
360
370
380
390
0
2
4
6
8
10
Time


GAMS
GAMS_PEN
BOBYQA
Concentration
0.09
0.10
0.11
0.12
0.13
0.14
0
2
4
6
8
10
time


GAMS
GAMS_PEN
BOBYQA
Temperature
0.73
0.74
0.75
0.76
0.77
0.78
0.79
0.80
0
2
4
6
8
10
time

Example 3: ASPEN Binary distillation column

Example 3: ASPEN Binary distillation column

- Process simulators which have embedded powerful first-order mathematical models.

Example 3: ASPEN Binary distillation column

- Process simulators which have embedded powerful first-order mathematical models.
- An explicit dynamic model of the underlying system and or its gradient are not available.

Example 3: ASPEN Binary distillation column

- Process simulators which have embedded powerful first-order mathematical models.
- An explicit dynamic model of the underlying system and or its gradient are not available.
- Limited optimization capabilities being most of them related to steady-state process simulation scenarios.

Example 3: ASPEN Binary distillation column

- Process simulators which have embedded powerful first-order mathematical models.
- An explicit dynamic model of the underlying system and or its gradient are not available.
- Limited optimization capabilities being most of them related to steady-state process simulation scenarios.
- Dynamic optimization algorithms are not available and are difficult to implement or are limited to a kind of linear MPC techniques.

Example 3: ASPEN Binary distillation column

- Process simulators which have embedded powerful first-order mathematical models.
- An explicit dynamic model of the underlying system and or its gradient are not available.
- Limited optimization capabilities being most of them related to steady-state process simulation scenarios.
- Dynamic optimization algorithms are not available and are difficult to implement or are limited to a kind of linear MPC techniques.
- A first-order dynamic model of a benzene/toluene binary distillation column was built using the ASPEN Dynamics simulation environment.

Example 3: ASPEN Binary distillation column

- Process simulators which have embedded powerful first-order mathematical models.
- An explicit dynamic model of the underlying system and or its gradient are not available.
- Limited optimization capabilities being most of them related to steady-state process simulation scenarios.
- Dynamic optimization algorithms are not available and are difficult to implement or are limited to a kind of linear MPC techniques.
- A first-order dynamic model of a benzene/toluene binary distillation column was built using the ASPEN Dynamics simulation environment.
- The equimolar feedstream is composed of a total flowrate of 1000 kmol/hr at 100 C and 2 bar. The distillation column has 6 trays and total condenser. Moreover, the column operates at 1 bar constant pressure and at reflux ratio of 1.4116.

Example 3: ASPEN Binary distillation column

- Process simulators which have embedded powerful first-order mathematical models.
- An explicit dynamic model of the underlying system and or its gradient are not available.
- Limited optimization capabilities being most of them related to steady-state process simulation scenarios.
- Dynamic optimization algorithms are not available and are difficult to implement or are limited to a kind of linear MPC techniques.
- A first-order dynamic model of a benzene/toluene binary distillation column was built using the ASPEN Dynamics simulation environment.
- The equimolar feedstream is composed of a total flowrate of 1000 kmol/hr at 100 C and 2 bar. The distillation column has 6 trays and total condenser. Moreover, the column operates at 1 bar constant pressure and at reflux ratio of 1.4116.
- The rigorous vapor-liquid equilibrium RadFrac module in Aspen Plus was used to perform the steady-state process simulation.

Example 3: ASPEN Binary distillation column

- Process simulators which have embedded powerful first-order mathematical models.
- An explicit dynamic model of the underlying system and or its gradient are not available.
- Limited optimization capabilities being most of them related to steady-state process simulation scenarios.
- Dynamic optimization algorithms are not available and are difficult to implement or are limited to a kind of linear MPC techniques.
- A first-order dynamic model of a benzene/toluene binary distillation column was built using the ASPEN Dynamics simulation environment.
- The equimolar feedstream is composed of a total flowrate of 1000 kmol/hr at 100 C and 2 bar. The distillation column has 6 trays and total condenser. Moreover, the column operates at 1 bar constant pressure and at reflux ratio of 1.4116.
- The rigorous vapor-liquid equilibrium RadFrac module in Aspen Plus was used to perform the steady-state process simulation.
- The steady-state simulation of the process was exported to obtain a dynamic model of the binary

  distillation column using the *Flow Driven* mode available in ASPEN.

Example 3: ASPEN Binary distillation column

- Process simulators which have embedded powerful first-order mathematical models.
- An explicit dynamic model of the underlying system and or its gradient are not available.
- Limited optimization capabilities being most of them related to steady-state process simulation scenarios.
- Dynamic optimization algorithms are not available and are difficult to implement or are limited to a kind of linear MPC techniques.
- A first-order dynamic model of a benzene/toluene binary distillation column was built using the ASPEN Dynamics simulation environment.
- The equimolar feedstream is composed of a total flowrate of 1000 kmol/hr at 100 C and 2 bar. The distillation column has 6 trays and total condenser. Moreover, the column operates at 1 bar constant pressure and at reflux ratio of 1.4116.
- The rigorous vapor-liquid equilibrium RadFrac module in Aspen Plus was used to perform the steady-state process simulation.
- The steady-state simulation of the process was exported to obtain a dynamic model of the binary

  distillation column using the *Flow Driven* mode available in ASPEN.

- Objective function: we use a MPC formulation for the objective function.

$$\min\ \Omega = \sum_{i=1}^{N}[(c_i - c^d)^2 + (u_i - u^d)^2] + \sum_{i=1}^{N}[\Delta c_i]^2 + \sum_{i=1}^{N}[\Delta u_i]^2$$

where c stands for the benzene mol fraction distillate stream which is the controlled variable, u stands for the reflux ratio which is the manipulated variable, N is the prediction and control horizons, the superscript d stands for desired or target values.

- Objective function: we use a MPC formulation for the objective function.

$$\min\ \Omega = \sum_{i=1}^{N}[(c_i - c^d)^2 + (u_i - u^d)^2] + \sum_{i=1}^{N}[\Delta c_i]^2 + \sum_{i=1}^{N}[\Delta u_i]^2$$

  where c stands for the benzene mol fraction distillate stream which is the controlled variable, u stands for the reflux ratio which is the manipulated variable, N is the prediction and control horizons, the superscript d stands for desired or target values.

- Two product dynamic transitions: (a) the initial steady-state is given by $[c_0, u_0]$=[0.8, 1.4116], whereas the target steady state is given by $[c^d, u^d]$=[0.9, 5.86] and (b) then the product transition in the opposite direction was done.

- Objective function: we use a MPC formulation for the objective function.

$$\min\ \Omega = \sum_{i=1}^{N}[(c_i - c^d)^2 + (u_i - u^d)^2] + \sum_{i=1}^{N}[\Delta c_i]^2 + \sum_{i=1}^{N}[\Delta u_i]^2$$

  where c stands for the benzene mol fraction distillate stream which is the controlled variable, u stands for the reflux ratio which is the manipulated variable, N is the prediction and control horizons, the superscript d stands for desired or target values.

- Two product dynamic transitions: (a) the initial steady-state is given by $[c_0, u_0]$=[0.8, 1.4116], whereas the target steady state is given by $[c^d, u^d]$=[0.9, 5.86] and (b) then the product transition in the opposite direction was done.
- We have assumed that on-line measurements of the benzene mol fraction in the distillate stream will be available every hour.

- Objective function: we use a MPC formulation for the objective function.

$$\min\ \Omega = \sum_{i=1}^{N}[(c_i - c^d)^2 + (u_i - u^d)^2] + \sum_{i=1}^{N}[\Delta c_i]^2 + \sum_{i=1}^{N}[\Delta u_i]^2$$

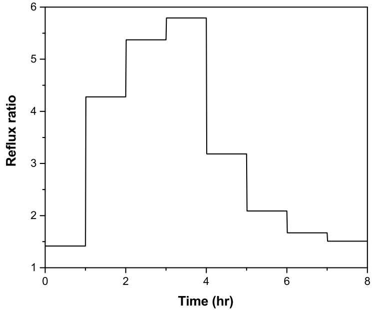
  where c stands for the benzene mol fraction distillate stream which is the controlled variable, u stands for the reflux ratio which is the manipulated variable, N is the prediction and control horizons, the superscript d stands for desired or target values.

- Two product dynamic transitions: (a) the initial steady-state is given by $[c_0, u_0]$=[0.8, 1.4116], whereas the target steady state is given by $[c^d, u^d]$=[0.9, 5.86] and (b) then the product transition in the opposite direction was done.
- We have assumed that on-line measurements of the benzene mol fraction in the distillate stream will be available every hour.
- For implementing both product dynamic transitions we proceeded as follows. (1) First, we initialize the simulation with the steady state conditions. (2) Then, change to dynamic mode and (3) run the dynamic simulation for 1 hr, (4) stop the simulation, (5) use the values of reflux ratio and concentration in the DFO BOBYQA solver as decision variables and solve the DFO problem; BOBYQA yields the new calculated value of the reflux ratio to be implemented in the simulation environment. Repeat steps (3)-(5) when new measurements become available until the target steady-state is reached.

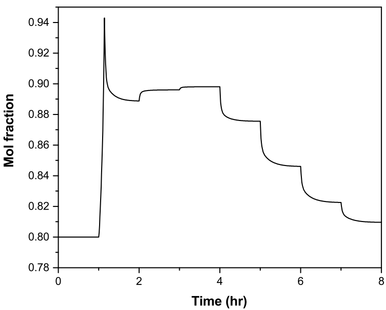
- Objective function: we use a MPC formulation for the objective function.

$$\min\ \Omega = \sum_{i=1}^{N}[(c_i - c^d)^2 + (u_i - u^d)^2] + \sum_{i=1}^{N}[\Delta c_i]^2 + \sum_{i=1}^{N}[\Delta u_i]^2$$

  where c stands for the benzene mol fraction distillate stream which is the controlled variable, u stands for the reflux ratio which is the manipulated variable, N is the prediction and control horizons, the superscript d stands for desired or target values.
- Two product dynamic transitions: (a) the initial steady-state is given by $[c_0, u_0]$=[0.8, 1.4116], whereas the target steady state is given by $[c^d, u^d]$=[0.9, 5.86] and (b) then the product transition in the opposite direction was done.
- We have assumed that on-line measurements of the benzene mol fraction in the distillate stream will be available every hour.
- For implementing both product dynamic transitions we proceeded as follows. (1) First, we initialize the simulation with the steady state conditions. (2) Then, change to dynamic mode and (3) run the dynamic simulation for 1 hr, (4) stop the simulation, (5) use the values of reflux ratio and concentration in the DFO BOBYQA solver as decision variables and solve the DFO problem; BOBYQA yields the new calculated value of the reflux ratio to be implemented in the simulation environment. Repeat steps (3)-(5) when new measurements become available until the target steady-state is reached.



Reflux ratio
Time (hr)



Mol fraction
Time (hr)

Conclusions

Conclusions

- We have carried out optimal product dynamic transition without the use of a explicit dynamic model.

Conclusions

- We have carried out optimal product dynamic transition without the use of a explicit dynamic model.
- Using a MPC framework it becomes feasible to compute on-line optimal transition policies.

Conclusions

- We have carried out optimal product dynamic transition without the use of a explicit dynamic model.
- Using a MPC framework it becomes feasible to compute on-line optimal transition policies.
- Future work: Dynamic transition of non-differentiable systems and solving MINLP problems using a DFO approach.

Conclusions

- We have carried out optimal product dynamic transition without the use of a explicit dynamic model.
- Using a MPC framework it becomes feasible to compute on-line optimal transition policies.
- Future work: Dynamic transition of non-differentiable systems and solving MINLP problems using a DFO approach.