# New Optimization Tools: The Julia Computing Environment

Antonio Flores-Tlacuahuac

School of Engineering and Sciences

Department of Chemical Engineering

ITESM-Monterrey

México

August 31, 2016

# Outline

- Introduction

# Outline

- Introduction
- JuMP

# Outline

- Introduction
- JuMP
- Optim

# Outline

- Introduction
- JuMP
- Optim
- StochJuMP

# Outline

- Introduction
- JuMP
- Optim
- StochJuMP
- NLopt

# Outline

- Introduction
- JuMP
- Optim
- StochJuMP
- NLopt
- Example: DFO of an ASPEN distillation column model

# Outline

- Introduction
- JuMP
- Optim
- StochJuMP
- NLopt
- Example: DFO of an ASPEN distillation column model

# Introduction

- Optimization as a tool for decision management

# Introduction

- Optimization as a tool for decision management
- Synonyms: Mathematical Programming, Operations Research

# Introduction

- Optimization as a tool for decision management
- Synonyms: Mathematical Programming, Operations Research
- Different approaches for system Optimization

# Introduction

- Optimization as a tool for decision management
- Synonyms: Mathematical Programming, Operations Research
- Different approaches for system Optimization
- Gradient based vs Free Derivative Algorithms

# Introduction

- Optimization as a tool for decision management
- Synonyms: Mathematical Programming, Operations Research
- Different approaches for system Optimization
- Gradient based vs Free Derivative Algorithms
- Some other approaches: Simulation and trial and error environments

# Introduction

- Optimization as a tool for decision management
- Synonyms: Mathematical Programming, Operations Research
- Different approaches for system Optimization
- Gradient based vs Free Derivative Algorithms
- Some other approaches: Simulation and trial and error environments
- Great Advances: From Unconstrained to PDE-based systems

# Introduction

- Optimization as a tool for decision management
- Synonyms: Mathematical Programming, Operations Research
- Different approaches for system Optimization
- Gradient based vs Free Derivative Algorithms
- Some other approaches: Simulation and trial and error environments
- Great Advances: From Unconstrained to PDE-based systems
- Computing power, Powerful algorithms, Automatic differentiation

# Introduction

- Optimization as a tool for decision management
- Synonyms: Mathematical Programming, Operations Research
- Different approaches for system Optimization
- Gradient based vs Free Derivative Algorithms
- Some other approaches: Simulation and trial and error environments
- Great Advances: From Unconstrained to PDE-based systems
- Computing power, Powerful algorithms, Automatic differentiation
- Algebraic Modeling Systems (AMPL, GAMS, AIMMS)

# Introduction

- Optimization as a tool for decision management
- Synonyms: Mathematical Programming, Operations Research
- Different approaches for system Optimization
- Gradient based vs Free Derivative Algorithms
- Some other approaches: Simulation and trial and error environments
- Great Advances: From Unconstrained to PDE-based systems
- Computing power, Powerful algorithms, Automatic differentiation
- Algebraic Modeling Systems (AMPL, GAMS, AIMMS)
- Pyomo (Python), JuMP (Julia)

# Introduction

- Optimization as a tool for decision management
- Synonyms: Mathematical Programming, Operations Research
- Different approaches for system Optimization
- Gradient based vs Free Derivative Algorithms
- Some other approaches: Simulation and trial and error environments
- Great Advances: From Unconstrained to PDE-based systems
- Computing power, Powerful algorithms, Automatic differentiation
- Algebraic Modeling Systems (AMPL, GAMS, AIMMS)
- Pyomo (Python), JuMP (Julia)

# Optim

- Mainly focused on solving unconstrained optimization problems with simple box constraints

# JuMP

- Algebraic Modeling System

# JuMP

- Algebraic Modeling System
- Hooked to some NLP and MINLP solvers

# JuMP

- Algebraic Modeling System
- Hooked to some NLP and MINLP solvers

# Example

Derivative-Free Optimization Approach for Product Transitions

- Traditional Optimal Model-based dynamic transitions

# Example

Derivative-Free Optimization Approach for Product Transitions

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

# Example

Derivative-Free Optimization Approach for Product Transitions

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

# Example

Derivative-Free Optimization Approach for Product Transitions

- 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

# Example

Derivative-Free Optimization Approach for Product Transitions

- 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

# Example

Derivative-Free Optimization Approach for Product Transitions

- 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
- Example

# Example

Derivative-Free Optimization Approach for Product Transitions

- 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
- Example
  - Binary distillation column in the ASPEN simulator

# Example

Derivative-Free Optimization Approach for Product Transitions

- 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
- Example
  - Binary distillation column in the ASPEN simulator

- 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.

- 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 $\Omega(\mathbf{y}, \mathbf{u})$.

- 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.

- 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.

- 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.

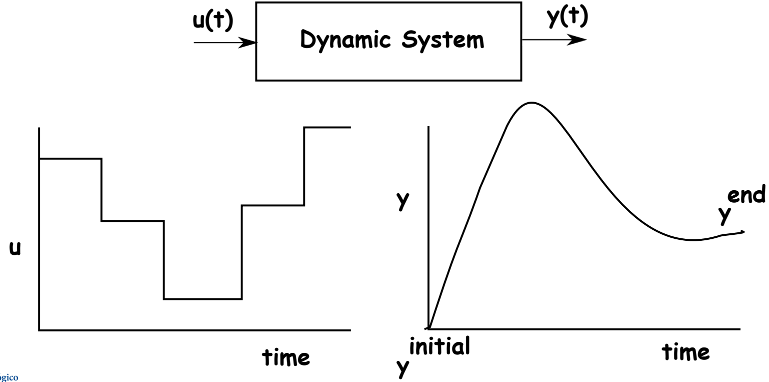
- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.
- The problem consists in computing the time domain values of the control variables $\mathbf{u}(t)$ such that the system response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

- 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.

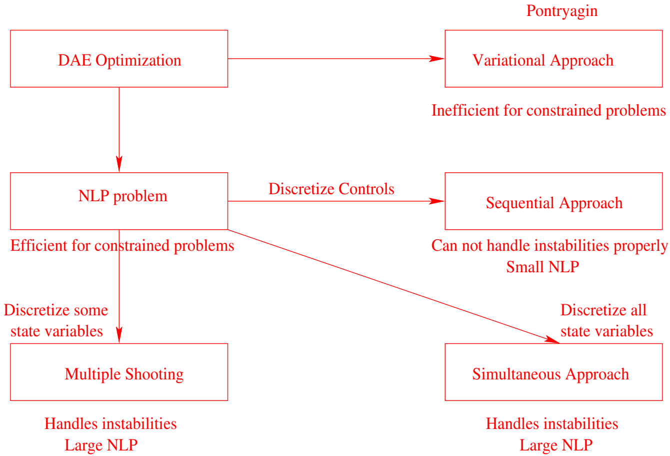
- During start-up, shut-down or normal operation, optimal dynamic transitions among products are normally required to achieve an operation target in the best possible way.
- The problem consists in computing the time domain values of the control variables $\mathbf{u}(t)$ such that the system response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

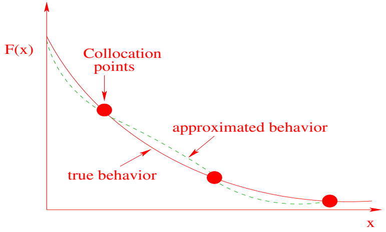
Given an ODE system:

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

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 $t_k$:

$$\begin{aligned} x_{k+1}(t) &= \sum_{k=0}^{\mathcal{K}} x_k \ell_k^x(t), \quad \ell_k^x(t) = \prod_{\substack{j=0 \\ j\neq k}}^{\mathcal{K}} \frac{t - t_j}{t_k - t_j} \\ u_k(t) &= \sum_{k=1}^{\mathcal{K}} u_k \ell_k^u(t), \quad \ell_k^u(t) = \prod_{\substack{j=1 \\ j\neq k}}^{\mathcal{K}} \frac{t - t_j}{t_k - t_j} \\ x_{N+1}(t_k) &= x_k, \quad u_N(t_k) = u_k \end{aligned}$$




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

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

$$\text{s.t.} \quad \frac{dx(t)}{dt} = F\left(x(t), u(t), t, p\right)$$

$$x(0) = x^0$$

$$g(x(t), u(t), p) \leq 0$$

$$h(x(t), u(t), p) = 0$$

$$x^l \leq x \leq x^u$$

$$u^l \leq u \leq u^u$$

Discretized NLP

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

$$\text{s.t.} \quad \sum_{j=0}^{\mathcal{K}} x_j \dot{\ell}_j(t_k) - F\left(x_k, u_k\right) = 0, \quad k = 1, \ldots, \mathcal{K}$$

$$x_0 = x(0)$$

$$g(x_k, u_k, p) \leq 0, \quad k = 1, \ldots, \mathcal{K}$$

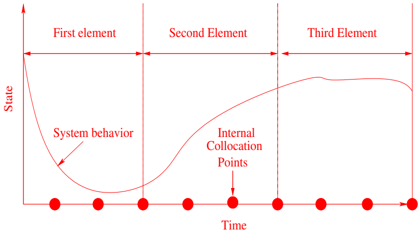
$$h(x_k, u_k, p) = 0, \quad k = 1, \ldots, \mathcal{K}$$

$$x^l \leq x \leq x^u$$

$$u^l \leq u \leq u^u$$

# 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

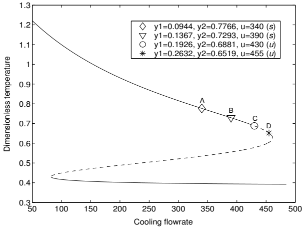
$$\frac{dC}{dt} = \frac{1 - C}{\theta} - k_{10} e^{-N/T} C$$

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

Parameter values

| | | |
|---|---|---|
| $\theta$ | 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 |
| $\alpha$ | $1.95 \times 10^{-4}$ | 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.

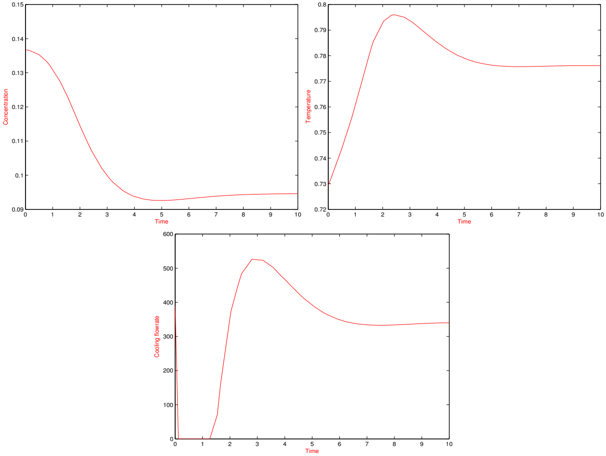
Tecnológico de Monterrey

julia

# 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.

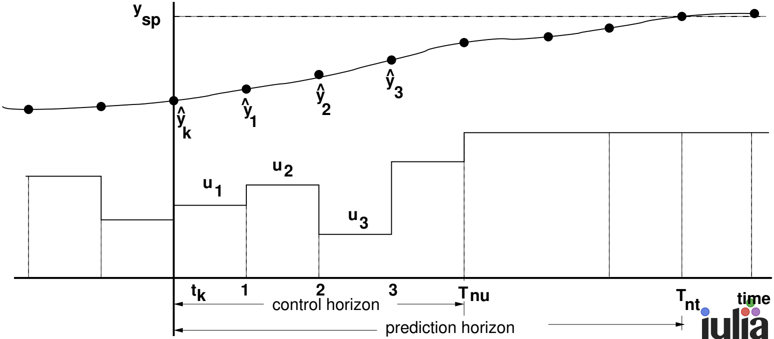
## Model Predictive control using black box dynamic modeling

- 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, $\lambda_x$, $\lambda_u$ 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, $\lambda_x$, $\lambda_u$ 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, $\lambda_x$, $\lambda_u$ 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:

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

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

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: ASPEN Binary distillation column

## Example: ASPEN Binary distillation column

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

## Example: 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: 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: ASPEN Binary distillation column

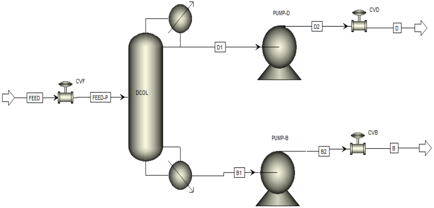
- Process simulators which have embedded powerful first-order mathematical models.
- An explicit dynamic model of the underlying system and or its gradient are not available.
- Limited optimization capabilities being most of them related to steady-state process simulation scenarios.
- Dynamic optimization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

## Example: 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.

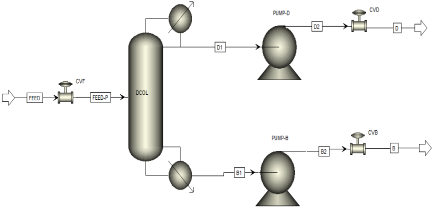
# Example: ASPEN Binary distillation column

# Example: ASPEN Binary distillation column



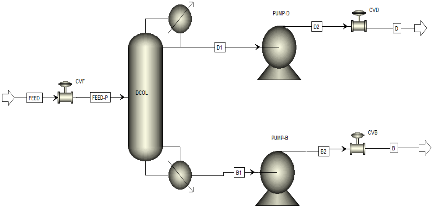

- A first-order dynamic model of a benzene/toluene binary distillation column was built using the ASPEN Dynamics simulation environment.

# Example: ASPEN Binary distillation column



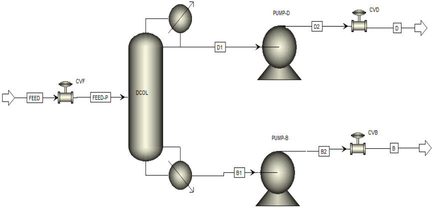

- A first-order dynamic model of a benzene/toluene binary distillation column was built using the ASPEN Dynamics simulation environment.
- The equimolar feedstream is composed of a total flowrate of 1000 kmol/hr at 100 C and 2 bar. The distillation column has 6 trays and total condenser. Moreover, the column operates at 1 bar constant pressure and at reflux ratio of 1.4116.

# Example: ASPEN Binary distillation column



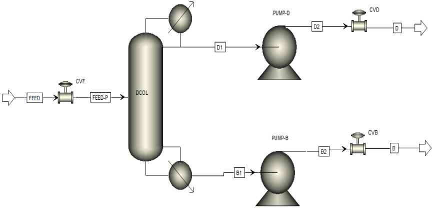

- A first-order dynamic model of a benzene/toluene binary distillation column was built using the ASPEN Dynamics simulation environment.
- The equimolar feedstream is composed of a total flowrate of 1000 kmol/hr at 100 C and 2 bar. The distillation column has 6 trays and total condenser. Moreover, the 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: ASPEN Binary distillation column




- 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: ASPEN Binary distillation column




- 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.

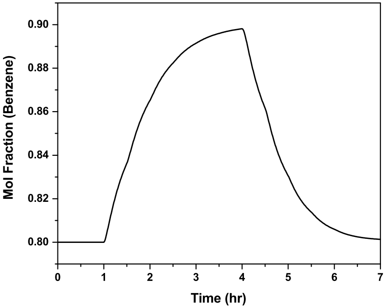
- Objective function: we use a MPC formulation for the objective function.

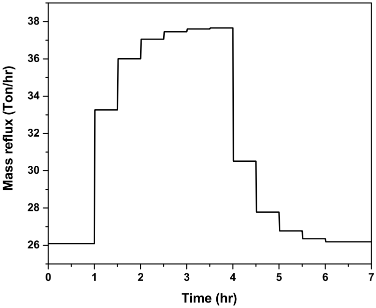
$$\min\ \Omega = \sum_{i=1}^{N}[(c_i - c^d)^2 + (u_i - u^d)^2] + \sum_{i=1}^{N}[\Delta c_i]^2 + \sum_{i=1}^{N}[\Delta u_i]^2$$

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.

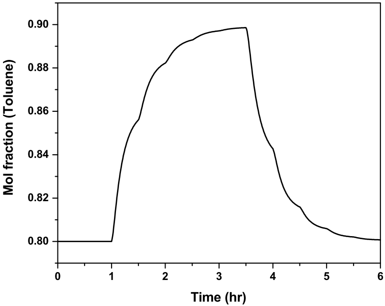
- Two product dynamic transitions: (a) the initial steady-state is given by $[c_0, u_0]$=[0.8, 1.4116], whereas the target steady state is given by $[c^d, u^d]$=[0.9, 5.86] and (b) then the product transition in the opposite direction was done.
- We have assumed that on-line measurements of the benzene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

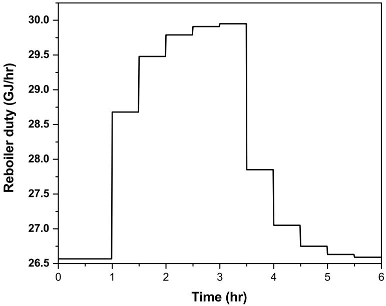
# One-point composition control of the distillate stream



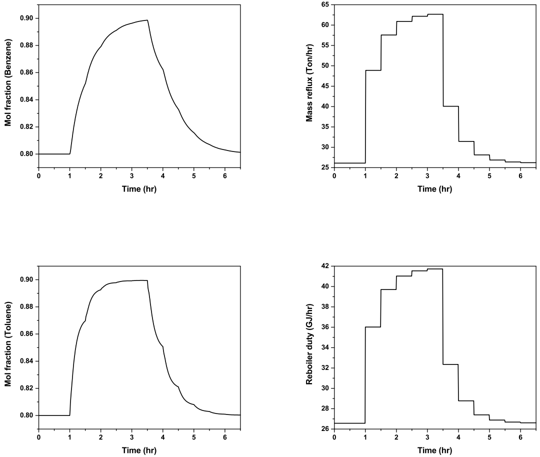



# One-point composition control of the bottoms stream

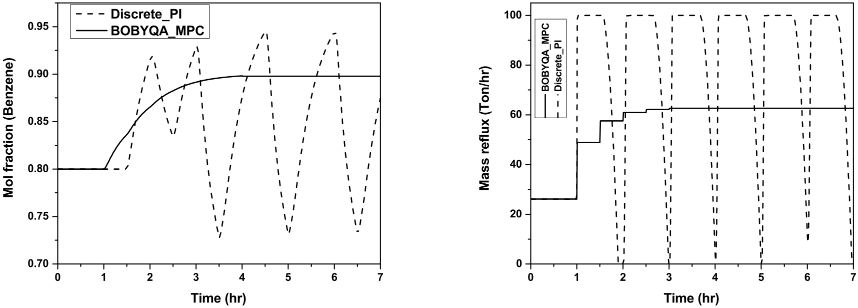


# Dual composition control of the distillate and bottoms streams



# Comparison against Proportional-Integral Controllers



## Conclusions

## Conclusions

- New, powerful optimization facilities available trough the Julia programming language

## Conclusions

- New, powerful optimization facilities available trough the Julia programming language
- Extension to solve numerical optimal control problems are required

# Conclusions

- New, powerful optimization facilities available trough the Julia programming language
- Extension to solve numerical optimal control problems are required
- We have carried out optimal product dynamic transition without the use of a explicit dynamic model.

# Conclusions

- New, powerful optimization facilities available trough the Julia programming language
- Extension to solve numerical optimal control problems are required
- 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

- New, powerful optimization facilities available trough the Julia programming language
- Extension to solve numerical optimal control problems are required
- 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

- New, powerful optimization facilities available trough the Julia programming language
- Extension to solve numerical optimal control problems are required
- 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.