# TÉCNICA DE IMPLEMENTAÇÃO

É implementada a técnica de *multipleshooting*, ou seja, as variáveis de decisão consideram não apenas as ações de controle , mas também os estados atuais e futuros. Assim, para o processo de otimização, ambas são vistas em todo o horizonte de predição () desejado.

Observar:

* Não se tem apenas uma ação de controle ótima, mas toda a trajetória ótima futura;
* é a condição/medição atual dos estados
* é a ação de controle atualmente aplicada ao processo controlado

Assim, considerando como sendo as variáveis de decisão[[1]](#footnote-1), e um estado atual diretamente medido do processo (representado por ) e ação de controle () atualmente na entrada da planta, a solução do NLP com a técnica *multipleshooting* pode ser escrita na forma:

sujeito às restrições de desigualdade que vão restringir o espaço de busca da solução:

|  |  |
| --- | --- |
|  | Eq. |

e outras restrições de igualdade:

* para “impor” a dinâmica futura do sistema a partir de uma condição inicial , conhecida pela medição atual
* para “impor” a variável de decisão em função das variáveis de decisão

|  |  |
| --- | --- |
|  | Eq. |

Importante notar que restrições de igualdade são “imposições”, ou seja, tem de respeitar o valor definido. As restrições de desigualdade associam faixas, ou seja, graus de liberdade que refletem o espaço de busca da solução.

Observar ainda que, para o caso em que se tem o modelo de equações que representam o sistema dinâmico, o estado futuro pode ser obtido na forma e as saídas controladas por *setpoint* são alguns dos estados que foram extraídos para estarem disponíveis na saída através da função . A função nada mais é senão um vetor/ matriz para seleção dos estados que vão ser apresentados na saída.

Para o caso em que usamos uma técnica de I.A. qualquer para representar o modelo dinâmico, os estados futuros serão preditos como uma função das ações de controle atuais aplicadas nos estados atuais do sistema: e as saídas controladas por *setpoint* são obtidas pela mesma função , mas agora representadas na forma de estimativas (acento circunflexo).

# OBJETOS:

Objetos definidos no CaSAdi, seja para a inicialização ou para contas durante a simulação.

|  |  |
| --- | --- |
| OBJ | DESCRIÇÃO |
| casadi\_solver | Criação do solver Casadi |
| Hp | Horizonte de predição |
| Hc | Horizonte de controle (HcHp) |
| nx | Número de variáveis de entrada (estados) do processo |
| ny | Número de variáveis (saída do processo) que são controladas por *setpoint* (no caso PChegada e Vazão) |
| nu | Número de variáveis manipuladas (ações de controle possíveis) - no caso, Freq e PMonAlvo |
| PassoMPC | Número de amostragens até a atuação do MPC (no caso = 3) |
| x0 | Condição inicial para os estados das variáveis medidas atuais e até o horizonte Hp (dimensão 1+Hp) |
| u0 | Condição inicial para guardar as ações de controle em todo o horizonte futuro (dimensão Hp) |
| BufferDeltaU | Para permitir somatório das últimas 15 variações na ação de controle |
| Predicao | Para guardar a predição no instante anterior |
| ModeloPreditor | Criação da variável para guardar modelo de preditor do processo e que será utilizada pelo solver para a predição |
| EstimaVazao | Função para carregar uma única vez a 'f\_Interpola\_casadi\_vazao\_sym' e para, com base na frequência e na PChegada (atual ou futura) poder proceder a estimativa da vazão (atual ou futura) |
| Funcao\_h | Para proceder a conta y=h(x) e obter as saídas em função da matriz h definida |
| lbx | Lower Bounds para as variáveis de decisão do MPC |
| ubx | Upper Bounds para as variáveis de decisão do MPC |
| lbg | Lower Bounds para as restrições [g] que forem criadas |
| ubg | Upper Bounds para as restrições [g] que forem criadas |
| contador | Variável para guardar o contador de passos de amostragem - usado para definir momentos de atuação do MPC |

# Formato CaSAdi

Para a implementação no modelo CaSAdi, os estados MPC devem ser colocados na forma [], onde:

* As variáveis de decisão do MPC precisam ser passados na forma de colunas, razão pela qual usamos a sintaxe (:)
* terá a dimensão de linhas e *1+* colunas (estado atual + estados futuros de até ). Assim, cada uma das colunas correspondente ao tempo de amostragem e cada uma das linhas corresponde a respectiva variável/estado;
* terá a dimensão linhas e colunas (ação atual + ações calculadas do horizonte até ). Assim, cada uma das colunas correspondente ao tempo de amostragem e cada uma das linhas correspondente a respectiva variável de entrada;
* terá a dimensão de linhas e colunas. Assim, cada uma das colunas correspondente ao tempo de amostragem e cada uma das linhas correspondente a respectiva variável de entrada;

# Formatação do problema de otimização

O problema de otimização do MPC deve atender as seguintes condições:

|  |  |
| --- | --- |
|  |  |
|  |  |
| ;  Para efeito de simulação, extrairemos condições reais do processo para serem usadas como condições iniciais  Já foi percebido que na maioria dos casos, inicializar uma variável de decisão com os valores futuros antes preditos, usualmente reduz o esforço computacional. Assim, a cada nova chamada solver, optamos por inicializar com valores antes preditos. O , por sua vez, pensamos em inicializar, mas concluímos ser desnecessário, uma vez que serão variáveis de decisão impostas por restrições de igualdade |  |

## Função custo

|  |  |
| --- | --- |
|  | Eq. |
|  |  |

Onde:

representa a diferença entre as saídas estimadas em todo e o valor de *setpoint* calculado pelo otimizador para as variáveis controladas por *setpoint*

corresponde ao erro atual entre as saídas medidas e a estimação das saídas feitas no instante anterior. Avaliado apenas no primeiro instante (não em todo horizonte futuro)

é a diferença entre a ação de controle aplicada na planta e os alvos definidos pela engenharia, em todo o

é a variação na ação de controle analisada em todo o futuro até

corresponde ao erro entre os estados medidos e a estimação dos estados feita no instante anterior. Avaliado apenas no primeiro instante (não em todo horizonte futuro)

## Variáveis de decisão do MPC e suas restrições em lbx/ubx:

É necessário lembrar que as variáveis de decisão para o solver do MPC estão na forma: [], onde

* terá a dimensão de [ *1+*, todos com limites em [LimitesMin , LimitesMAx]\*;
* terá a dimensão de  \*\*, todos com limites em [, ]\*\*\*;
* terá a dimensão de , todos com limites [-, ]

\* LimitesMin e LimitesMax foram definidos na inicialização apenas para restringir o espaço de busca. Para isso usamos como referência os limites de alarme de cada uma das variáveis, com uma margem de folga de 10%. Observar ainda que os estados (variáveis do processo) e mesmo as saídas que nada mais são senão parte dos estados, ambas têm limites que mudam em função da frequência (atual e futura), neste caso, as restrições para estas variáveis serão tratadas em lbg/ubg e por isso foram “liberadas” em lbx/ubx.

\*\* Importante observar que em sendo , será necessário calcular a ação de controle ótima até o horizonte . Por outro lado, em atendimento a teoria, a ponderação das ações de controle na função custo deve considerar apenas até o horizonte . O mesmo ocorrerá com o , o qual vai compor a função custo apenas até

\*\*\* Observar que a PMonAlvo, na prática, é a proposição para a PChegada. A PChegada, por sua vez, tem restrições que são função da frequência (atual e futura). Assim, é importante que as restrições da entrada *U* associadas a PMonAlvo (atuais e futuras), sejam também avaliadas em função das restrições da PChegada. Isso será tratado nas restrições em lbg/ubg.

**OBSERVAÇÃO:**

Fizemos experimentos na tentativa de tratar o por faixas, ou seja,max>min>0, o que corresponderia a busca do solver as faixas [-max até - min], [zero], [min até max]. Não vale a pena pois esta descontinuidade viola condições do solver, o qual assume a premissa de que as restrições são diferenciáveis, portanto, não devem existir descontinuidades no espaço de busca da solução.

## Restrições de igualdade em lbg/ubg

Observe que temos a função do estimador que nos oferece e as saídas controladas por *setpoint* são obtidas pela função . Assim, dada uma condição atual na entrada do processo e uma ação de controle atualmente aplicada, podemos estimar .

O papel do otimizador começa na busca da primeira ação de controle ótima () que atenderá as restrições definidas. Uma vez conhecido o estado futuro e a ação de controle ótima futura, é possível estimar um estado futuro para o passo seguinte . A sequência até o horizonte de predição desejado[[2]](#footnote-2), nos leva a completar a seguinte tabela:

|  |  |  |  |
| --- | --- | --- | --- |
| Estado  Dimensão = | Ação Atual  Dimensão = | Dimensão = | Instante k futuro |
|  |  |  | k=0 (atual) |
|  | Cálculo de |  | k=1 |
|  | Cálculo de |  | k=2 |
|  |  |  |  |
|  | Cálculo de |  | k=Hp-1 |
|  |  |  | k=Hp |

Como comentado pela técnica do *multipleshooting*, as restrições de igualdade devem impor a dinâmica (ver Eq. 2). Da mesma forma, serão usadas restrições de igualdade para impor e deixar apenas as ações de controle como reais variáveis de decisão. Assim, observada a primeira e a terceira colunas da tabela anterior, podemos escrever as restrições de igualdade na forma:

## Restrições de desigualdade em lbg/ubg

Apenas para efeito de esclarecimento, quando usamos a expressão , é importante entender que estamos nos referindo ao instante atual e a qualquer outro instante futuro predito. Isso é importante, por exemplo, para entender que as ações de controle futuras também precisam considerar as restrições dinâmicas futuras, que dependem da própria ação de controle (especialmente frequência) futura. Da mesma forma, a estimação da vazão é feita com base na Frequência e na PChegada, portanto, estados futuros da vazão dependem de valores futuros da Frequência e da PChegada e as restrições futuras da futura vazão também precisam estar contempladas.

* + Os estados preditos (atuais e futuros) precisam estar dentro dos limites operacionais calculados em função da frequência, ou seja ,
  + As saídas controladas por *setpoint* (atuais e futuras) precisam estar dentro dos limites operacionais calculados em função da frequência, ou seja, ,
  + As variáveis de entrada a serem aplicadas no sistema (ações de controle que são Frequência e PMovAlvo, representadas como []), precisam respeitar os limites operacionais, ou seja,
  + A ação de controle correspondente a PMonAlvo () precisa respeitar os limites dinâmicos correspondentes a PChegada, que mudam em função da frequência, ou seja,
  + A variação no esforço de controle a ser aplicado, precisa respeitar os limites operacionais, ou seja, , sendo
  + Na ação de controle correspondente frequência, definida como , a variação máxima deve respeitar limites definidos pelo fabricante. No caso do JUB27, no máximo 1Hz a cada 7,5min

**Observação 1:**

Os limites mínimos e máximos das variáveis do processo são calculados de forma dinâmica, em função da frequência. Estes limites correspondem a valores de alarmes L e H definidos pela empresa. Considerando que há alarmes que podem causar trip da planta, a implementação do código deve considerar a possibilidade de uma margem de tolerância percentual definida pelo usuário, tal qual indica a formulação seguinte. A definição de uma *MargemPercentual* = ZERO faz com que os limites originalmente calculados sejam mantidos. Sugerimos atuar com *MargemPercentual* = 1.

**Observação 2:**

Para implementar a última restrição, sabemos que a variação máxima permitida para a frequência é de 1Hz (MaxDeltaHz=1) a cada 7,5min (TempoLimite=450s).

Das configurações originais do sistema, temos um tempo de amostragem (Ts) = 10s, de modo que os 450s correspondem a 45 amostragens.

Observar que existem duas ações de controle. Nesta formulação representaremos a ação correspondente a frequência pela variável e sua variação é dada por

Para implementação no instante *k*, teremos de ofertar ao controlador, um vetor com as últimas 44 atuações do controlador[[3]](#footnote-3). Para calcular o , o otimizador deve então considerar as seguintes restrições:

Importante lembrar que esta restrição vale , ou seja, precisará ser vista para todo o horizonte futuro das ações de controle preditas, independentemente do tamanho do horizonte futuro.

# Implementação em código

## Obter dados no instante atual do processo

O bloco CaSAdi deve receber:

* = medições dos estados atuais da planta
* = Informação atual da ação de controle que foi efetivamente aplicada na planta
* Alvos de engenharia definidos pelo usuário (Freq e PMonAlvo)

## Variáveis de decisão para o solver

A inicialização das variáveis de decisão para o solver será: [ ], em formato de coluna única

* terá a dimensão de [
* terá a dimensão de
* terá a dimensão de

A dimensão das variáveis de decisão será então: , lembrando que pelo método *multipleshooting*, a variável é tratada com restrições de igualdade, ou seja, não requer busca do solver e as variáveis de decisão efetivamente utilizadas são as ações de controle em todo o horizonte . Também vale registrar que, sem perda de generalidade, podemos inicializar com zeros, já que, apesar de serem variáveis de decisão, não haverá busca destas variáveis (atuais e futuras), uma vez que vão ser valores impostos por restrições de igualdade.

**IMPORTANTE:**

* Tentamos usar apenas ou apenas como variável de decisão, mas não foi possível tratar de forma isolada. Isso ocorre pelo fato de que precisamos de decisões que restrinjam diretamente o em limites máximos, assim como, as restrições dinâmicas futuras precisam da variável de decisão em , atual e futura. Neste contexto, entendemos ser necessário passar ambas como variáveis de decisão. Por outro lado, como há uma relação direta entre elas, foi tratada com restrições de igualdade
* No código implementado, para padronizar e facilitar o entendimento de todos, AS INFORMAÇÕES NO TEMPO ESTARÃO EM COLUNAS, ou seja, as linhas representam as variáveis e cada coluna da matriz corresponderá ao respectivo estado atual/futuro destas variáveis.

## Parâmetros para serem enviados para o solver

Para cada chamado do solver, teremos de atualizar os parâmetros que lhe serão enviados, e que deverão compor um vetor coluna na forma:

Onde:

* deve conter a medição atual das variáveis do processo (dimensão )
* deve conter a ação de controle atualmente aplicada na planta (dimensão )
* é o alvo correspondente ao ponto de operação definido pela engenharia (valor para a Frequência e PMonAlvo desejados) (dimensão )
* são as saídas consideradas ótimas para as variáveis controladas por *setpoint* (PChegada e Vazão)
* corresponde ao erro entre os estados medidos e a estimação dos estados feita no instante anterior (dimensão )
* corresponde ao erro atual entre as saídas medidas e a estimação das saídas feitas no instante anterior. (dimensão )
* tem dimensão 45 e traz o valor de atual proposto além dos 44 últimos valores de efetivamente aplicados ao processo
* são os estados do reservatório da ESN que precisam ser atualizados a cada passo de amostragem para que a ESN não se perca ao longo do tempo. (dimensão que corresponde ao tamanho do reservatório da ESN utilizada)

OBSERVAÇÃO:

O solver vai calcular novas ações de controle ótimas, ou seja, Frequência e PMonAlvo ótimas para serem aplicadas no processo. Assim, vamos assumir que o valor Ysp ótimo para a saída referente a PChegada é o valor da PMonAlvo ótima calculada pelo solver, assim como, para a saída que se refere a Vazão, vamos assumir que a vazão ótima é a vazão estimada para o ponto da Frequência PMonAlvo ótimos dados pelo solver. Para completar a informação do ótimo nos três mapas, precisaremos também da PSuc ótima. Neste caso, assim como a vazão, a PSuc será estimada no ponto da Frequência PMonAlvo ótimos dados pelo solver.

1. Observar que é a entrada atual da planta passado para o otimizador, assim como foi o . As medições são usadas para impor a técnica multipleshooting e o não muda, mas vale observar que , enquanto variável de decisão, vai ser alterada pelo solver, assim como todos os !! [↑](#footnote-ref-1)
2. O texto ilustra a sequência para as predições e ações de controle futuras, no entanto, é importante lembrar que pela técnica multishooting, não é feito um passo por vez, mas todos os passos de uma vez só !! [↑](#footnote-ref-2)
3. Apesar do controlador só atuar de 30 em 30s, optamos por fazer a conta a cada amostragem de 10s, mesmo sendo o caso de somar . Isso, pois, nos casos em que o controlador der *unfeasible*, ele tentará uma nova ação no ciclo seguinte. Assim, não há sincronismo com os 30s, mas sim com o período de amostragem de 10s. [↑](#footnote-ref-3)