1 Desenvolvimento de Aeronaves por Programação Geométrica

1.1 Introdução

Neste capítulo vamos ver como podemos usar um pouco do que vimos até agora para resolvermos um problema um pouco mais concreto do que os apresentados como exemplos. O problema a ser analisado é o de projeto (*design*) de aeronaves.

Pelas últimas décadas, métodos de otimização têm tomado cada vez maior importância no desenvolvimento de máquinas complicadas, mas ainda com muitas limitações. O problema é que, quando um projeto é tão grande e complicado quanto o do desenvolvimento de aeronaves, que contam com relações altamente não lineares entre muitas variáveis, a aplicação de um método homogêneo, ou mesmo de vários métodos heterogêneos, fica difícil. Não é pouco usual a aplicação de um método de otimização a uma parte específica do projeto (o da asa, por exemplo) e deixá-lo rodando por dias para obter, no final, apenas uma resposta aproximada, que pode ou não ser viável quando se leva em consideração o resto do projeto.

Para estudar como melhor lidar com esse tipo de situação, surgiu o campo de estudos de *Otimização Multidisciplinar de Projetos* (ou *Multidisciplinary Design Optimization*, **MDO** na sigla em inglês, que será usada daqui para frente). **MDO** O objetivo dos métodos MDO são coordenar de maneira eficiente diferentes métodos de otimização para um projeto único, na esperança de assim obter projetos melhores ("melhores" nos termos definidos pelo ou pela projetista) do que os obtidos otimizando o projeto por partes isoladas. No entanto, mesmo depois de numerosos avanços em métodos MDO, aplicá-los a um projeto inteiro de uma aeronave ainda é impraticável na maior parte dos casos.

Em última instância, a utilização de métodos de otimização para esse tipo de problema (o de projetos como o de uma aeronave, como no exemplo) é feita a fim de se entender e analisar melhor o problema em mãos. Isso porque, no início, o problema a ser resolvido pelo projeto não é, no geral, bem posto (muitas vezes, os objetivos do projeto não são sequer definidos[1]), e é de interesse entender o formato de sua fronteira de Pareto[1], a fim de se entender a relevância de cada característica (ou, poderíamos dizer, "de cada variável"). Isso, aliado ao fato de que geralmente um projetista gostaria de otimizar vários parâmetros, e não só um, aumenta a necessidade de se obter não só uma "resposta" otimizada, mas também um maneira de entender a vizinhança dessa resposta (isto é, realizar uma análise de sensitividade).

Vale também notar que há um impasse entre a realização de uma análise

[1] Uma alocação de recursos (codificada aqui como a atribuição de valores a variáveis) é dita Pareto eficiente se não é possível alterar tal alocação tal que um critério de otimalidade seja melhorado sem que outro seja piorado. Fronteira de Pareto é o conjunto de alocações Pareto eficientes.

de alta ou de baixa fidelidade. Para análises de alta fidelidade, existem duas opções:

- Fazer uma análise de uma parte específica (como o da asa, ou do motor) do projeto, o que possibilitaria a quem cuidar dele fazer decisões mais informadas sobre essa parte específica;
- Fazer uma análise mais geral, envolvendo um subconjunto maior de "partes específicas", incorrendo em instâncias mais arbitrárias de problemas mais gerais, e leva, em geral, dias ou semanas para a obtenção de uma solução.

Esses tipos de problemas, assim como a possibilidade de *overfitting* [2] levam, em vários casos, a uma escolha por uma análise de baixa fidelidade.

Antes de seguirmos um pouco mais a fundo, convém obtermos uma ideia geral do desenvolvimento típico de uma aeronave. Esse desenvolvimento é composto por três etapas[1] (ou, pelo menos, assim os engenheiros o têm tratado por algumas décadas):

- Projetagem conceitual: etapa em que os objetivos do projeto são definidos, assim como requisitos de performance. Dependendo do projeto, esta etapa pode levar vários anos e pode envolver uma análise matemática razoavelmente detalhada;
- Projetagem preliminar: envolve uma análise mais aprofundada da configuração do projeto, identificação e resolução de problemas de interferência aerodinâmica e de instabilidade é realizada e, em um certo ponto, a configuração básica da aeronave é "congelada", de modo a permitir que times que trabalham com subsistemas possam trabalhar independentemente, sem afetar demais os outros times;
- Projetagem detalhada: envolve uma projetagem detalhada do projeto, desenhos em CAD, determinação de passagens hidráulicas, elétricas e de combustível, assim como fabricação de peças de produção[3].

Cada uma dessas etapas pode envolver grandes problemas de otimização com restrições, envolvendo as dificuldades notadas acima.

1.2 Programação Geométrica

Nos últimos anos, técnicas de programação convexa têm se tornado cada vez mais eficientes (aproximando mesmo técnicas de programação linear). Apesar disso, por incrível que possa parecer, tais técnicas ainda são notavelmente

[2]Uma solução "está em" *overfitting* quando levar a excelentes resultados para uma comparativamente pequena quantidade de casos de teste, mas a soluções ruins quando em situações mais gerais.

[3]Peças que são pensadas para uso em um projeto específico.

ausentes em propostas MDO[1] e a abordagem mais comum a esses problemas é por meio de métodos gerais de programação não-linear. Assim, problemas muito gerais podem ser abordados, mas dificilmente os critérios de confiança e e eficiência são preenchidos.

No lugar disso, então, voltamos nossa atenção a métodos de otimização convexa e, em particular, a um método que se mostrou particularmente adequado para lidar com problemas de projeto de aeronave, o de programação geométrica[4]. Quando esse método é aplicável, entre as vantagens que ele oferece estão a obtenção de soluções globalmente ótimas (ou um certificado de infactibilidade, quando não existem tais soluções), com tempos de solução comparativamente curtos, que escalam para problemas maiores (isto é, quando o problema aumenta "um pouco", o tempo de solução aumenta só "um pouco"). Ele tem a desvantagem de não ser um método geral, mas como só estamos interessados (no momento) em resolver problemas de projetagem de aeronaves, essa desvantagem não nos entristece muito.

Um problema de programação geométrica em forma padrão pode ser posto da seguinte forma:

$$
\begin{aligned}
minimize\ & \sum_{i=1}^{n} f_0(x) \\
tal\ que\ & f_i(x) \leq 1, i = 1, ..., n, \\
& g_i(x) = 1, i = 1, ..., k, \\
& x \in R^n_+,
\end{aligned}
$$

em que f_i são posinômios ou monômios, os g_i são monômios e $x \in R^n$. Aqui, a definição de monômios (e de posinômios) é um pouco diferente do usual em álgebra:

- Um **monômio** $g(x)$ é algo da forma $g(x) = kx^\alpha$, $\alpha \in R^n, k \in R^n_+$[5]; **monômio**
- Um **posinômio** é algo da forma $f(x) = \sum g_i(x)$, em que cada g_i é um monômio (em particular, monômios são posinômios). **posinômio**

Note que esse problema não só é altamente não-linear, mas também é não-convexo. Nós podemos, no entanto, torná-lo convexo, com uma mudança de coordenadas apropriada. Note que, se f é um posinômio em x, $log(f(e^x))$ é convexo e, se g é um monômio em x, $log(g(e^x))$ é, não só convexo, mas também afim. Realizando essa mudança de variáveis, nosso problema de otimização pode ser expresso como

[4] Diferentemente do que se pode supor, o nome "programação geométrica" não é devido às suas propriedades geométricas no geral, mas sim à desigualdade geométrico-aritmética, que foi muito usada em sua análise.

[5] Aqui, usamos a convenção de que, se $x \in R^n$ e $\alpha \in R^n$, $x^\alpha := \prod x_i^{\alpha_i}$.

$$
\begin{aligned}
minimize\ log \sum e^{\alpha'_{0k}x+b_{0k}} \\
tal\ que\ log \sum e^{\alpha'_{ik}+b_{ik}} \leq 0 \\
x \in R^n_+.
\end{aligned}
$$

Nessa nova formulação, estamos tomando α como a matriz de vetores que geram as potências dos monômios e posinômios. Note que, fazendo essa mudança de variáveis ($v = log(x)$), monômios se tornam funções afim e, posinômios, funções convexas.

É importante notar que monômios e posinômios são um conjunto fechado sob as operações de divisão por monômios, então desigualdades do tipo $f_1(x) = f_2(x)$, em que f_2 é um monômio e f_1 é um posinômio poderiam ser lidadas fazendo $\frac{f_1(x)}{f_2(x)} = 1$. Ademais, igualdades como $g(x) = 1$, se g é um monômio, podem ser lidadas fazendo $g(x) \leq 1$ e $\frac{1}{g(x)} \leq 1$[6]. A restrição de que k precisa ser positivo acaba sendo frequentemente satisfeita sem esforços mas, quando esse não é o caso, é preciso se usar outro maquinário que não o da programação geométrica. Em particular, programação signomial[7], que faz uso de uma estrutura parecida com a da geométrica, resolve esse caso (no geral, de forma mais eficientemente do que um método de programação não linear genérico).

1.3 Um modelo simples de aeronave

A seguir é apresentado um modelo de aeronave simplificado, em grande parte baseado no modelo da tese de Hoburg[1]. A modelagem do problema será feita com base no pacote GPkit.

1.3.1 Modelo de Peso e Sustentação

Assumimos que a aeronave esteja em vôo de cruzeiro[8], de modo que o sustentação seja igual ao arrasto e a sustentação igual ao peso. O peso da aeronave é a soma do peso de carga útil P_0, o da asa P_a e do combustível P_c:

$$P \geq P_0 + P_a + P_c$$

Usamos o modelo abaixo para vôo estável[9]:

$$P_0 + P_a + \frac{P_c}{2} \leq \frac{1}{2}\rho S C_l V^2$$

[6]Em particular, isso mostra que um programa geométrico qualquer pode ser expresso com desigualdades do tipo $f_i(x) \leq 1$, o que é a entrada que alguns *solvers* requerem por padrão.

[7]Um posinômio é um signômio que é restrito a ter coeficientes positivos. Programação signomial é a cujas restrições são signômios.

[8]Quando as velocidades linear a angular da aeronave são constantes em um sistema de referência fixado ao corpo. É o mesmo que "vôo estável".

[9]O "l" abaixo vem de "*lift*", "sustentação" em inglês.

em que a sustentação da aeronave é igual ao peso dela com metade de combustível, C_l é o coeficiente de sustentação e V a velocidade do avião.

Aqui usamos a equação para arrasto:

$$Arrasto = D := \frac{1}{2}\rho S C_d V^2$$

em que $\frac{1}{2}\rho V^2$ representa a pressão dinâmica, S corresponde à área da asa e C_d ao coeficiente de arrasto total[10], dado por:

$$C_{d_{fuse}} = \frac{CDA_0}{S} + kC_f \frac{S_{wet}}{S} + \frac{C_l^2}{\pi A e}$$

em que CDA_0 corresponde à área de arrasto da fuselagem[11], k é um fator de forma que leva em conta arrasto de pressão, $\frac{S_{wet}}{S}$ é a razão de área molhada[12] e e é o fator de eficiência de Oswald. CDA_0 é linearmente proporcional ao volume de combustível na fuselagem[13]:

$$V_{f_{fase}} \leq CDA_0 \times 10[metros]$$

Assumindo um fluxo de Blasius turbulento sobre uma placa plana, o coeficiente de fricção C_f pode ser aproximado por:

$$C_f = \frac{0.074}{Re^{0.2}}$$

em que $Re = \frac{\rho V}{\mu}\sqrt{\frac{S}{A}}$ é o número de Reynolds no meio da corda $\sqrt{\frac{S}{A}}$.

Também gostaríamos que uma aeronave cheia de combustível seja capaz de voar a uma velocidade mínima:

$$P \leq \frac{1}{2}\rho V_{min}^2 S C_{l_{max}}$$

em que $C_{L_{max}}$ é o coeficiente de sustentação máximo.

Uma medida de performance útil, o tempo de vôo, é dada pela distância percorrida sobre velocidade:

$$T_{vôo} \geq \frac{Distância}{V}$$

A razão de força de sustentação/arrasto é dada por:

$$\frac{L}{D} = \frac{C_l}{C_d}$$

[10] O "d" vem de "*drag*", arrasto em inglês.

[11] Definida como a área normal ao fluxo de ar que geraria o mesmo arrasto que o avião.

[12] A área em contato com o fluxo de ar.

[13] GPkit faz checagem de unidades, então precisamos corrigi-la explicitamente, do que o "[metros]" constitui um lembrete.

em que C_l é o coeficiente de sustentação.

Essas restrições se traduzem como

```
# Modelo de peso e empuxo
restricoes += [P >= P_0 + P_a + P_f,
               P_0 + P_a + 0.5 * P_f <=
               0.5 * rho * S * C_l * V ** 2,
               P <= 0.5 * rho * S * C_lmax * V_min ** 2,
               T_voo >= Dist / V,
               LoD == C_L/C_D]
```

Assumimos que o Consumo Específico de Combustível por Unidade de Empuxo (CEUE) é constante e que ele provê tanto impulso quanto necessário. Como a força de impulso F é maior ou igual que a força de arrasto D, obtemos

$$P_f \geq CEUE \times T_{vôo} \times D$$

E podemos completar nosso modelo da seguinte forma:

```
# Modelo de Impulso e Arrasto
C_D_fuse = CDA0 / S
C_D_wpar = k * C_f * S_wetratio
C_D_ind  = C_l ** 2 / (np.pi * A * e)
restricoes += [P_f >= CEUE * T_voo * D,
               D >= 0.5 * rho * S * C_D * V ** 2,
               C_D >= C_D_fuse + C_D_wpar + C_D_ind,
               V_f_fuse <= 10*units('m')*CDA0,
               Re <= (rho / mu) * V * (S / A) ** 0.5,
               C_f >= 0.074 / Re ** 0.2]
```

1.3.2 Volume do Combustível

Definindo o volume requerido em função da densidade do combustível ρf, temos

$$V_c = \frac{P_c}{\rho_f g}$$

Pelo modelo aerodinâmico, temos que

$$V_{c_{asa}}^2 \leq 0,0009 \frac{S\tau^2}{A \times Distância}$$

O volume de combustível disponível $V_{c_{disp}}$ é limitado superiormente pelos volumes disponíveis na asa e na fuselagem:

$$V_{c_{disp}} \leq V_{c_{asa}} + V_{c_{fuse}}$$

Note que, devido à restrição acima, o programa resultante será signomial.

Restringimos o volume total de combustível como menor do que o volume disponível:

$$V_{c_{disp}} \geq V_c$$

```
# Modelo de Volume de Combustível
with SignomialsEnabled():
    restricoes +=[V_c == P_c / g / rho_c,
                  V_c_wing**2 <= 0.0009*S**3/A*tau**2,
                  # linear em b e tau, quadratico em chord
                  V_c_avail <= V_c_wing + V_c_fuse, #[SP]
                  V_c_avail >= V_c]
```

1.3.3 Acumulação de Peso nas Asas

O peso na superfície da asa é uma função da área da asa

$$P_{p_{surp}} \geq P_{p_{coef2}} S$$

O peso estrutural na asa é uma expressão posinomial que leva em conta o alívio de cisalhamento devido à presença de combustível nas asas e o momento flexor:

$$P_{p_{strc}}^2 \geq \frac{P_{p_{coeff1}}^2}{\tau^2} (N_{ult}^2 AR^3((P_0 + \rho_f g V_{c_{fuse}}) PS))$$

O peso total da asa é limitado:

$$P_p \geq P_{p_{surp}} + P_{p_{strc}}$$

```
# Modelo de Peso nas Asas
restricoes += [P_w_surf >= P_P_coeff2 * S,
               P_w_strc**2. >= P_P_coeff1**2./ tau**2. *
               (N_ult**2. * A ** 3. * ((P_0+V_f_fuse*g*rho_f) * P * S)),
               P_w >= P_w_surf + P_w_strc]

return restricoes
```

1.3.4 Possíveis Objetivos

Dadas essas restrições, algumas potenciais funções objetivo com as quais poderíamos querer trabalhar são:

- P_c, o peso do combustível (o objetivo padrão);
- P, o peso total da aeronave;

- $\frac{P_c}{T_{vôo}}$, o peso do combustível dividido pelo tempo de vôo (podemos pensar nisso como uma medida de eficiência no uso do combustível);
- $P_c + c \times T_{vôo}$, uma combinação linear antre peso do combustível e tempo de vôo.

Leituras adicionais

[1] Hoburg, W. Warren, “Aircraft Design Optimization as a Geometric Program” 2013.

[2] E. Burnell and W. Hoburg, “Gpkit software for geometric programming.” https://github.com/convexengineering/gpkit, 2018. Version 0.7.0.

[3] B. Ozturk, “SimPleAC” https://github.com/convexengineering/gplibrary/blob/master/gpkitmodels/SP/SimPleAC/simpleac.pdf, 2018.