1 Optimization Problem

Problem is

$$\min_{x \in X} f_0(x), \tag{1}$$

$$f_i(x) \leq 0, \tag{2}$$

for $i = 1, \ldots, m$, where $X \subseteq \mathbb{R}^n$ is a box:

$$\{x \mid x_j^{\min} \leq x_j \leq x_j^{\max}\}. \tag{3}$$

2 Approximate Problem

Suppose kth iterate is $x^{(k)}$. Then, for $i = 0, \ldots, m$, replace $f_i(x)$ with

$$g_i(x) = f_i(x_0) + \nabla f_i(x_0) \cdot (x - x^{(k)}) + \frac{\rho_i^2}{2} \left| \frac{x - x^{(k)}}{\sigma} \right|^2, \tag{4}$$

where σ and ρ are vectors. And make a trust region T (actually it's $T \cup X$)

$$T = \{x \mid |x_j - x_j^{(k)}| \leq \sigma_j\}. \tag{5}$$

So that the new problem is

$$\min_{x \in T} g_0(x), \tag{6}$$

$$g_i(x) \leq 0. \tag{7}$$

3 Overall Scheme

For the kth iteration:

1. Solve approximate problem to find candidate $x^{(k+1)}$
2. Check conservative: $g_i(x^{(k+1)}) < f_i(x^{(k+1)})$.
   - If no, throw away candidate, double ρ_i for each non-conservative g_i, and solve approximate problem again.
3. Halve ρ (take bigger steps) and update σ (decrease σ_i if x_i oscillating, increase if monotonic i.e. heading somewhere else).

4 Solving approximate problem

4.1 Evaluating dual function

Lagrangian relaxation, where $\lambda_0 = 1$,

$$L(x, \lambda) = \sum_{i=0}^{m} \lambda_i g_i(x) \tag{8}$$

$$= \sum_{i=0}^{m} \lambda_i f_i(x_0) + \left(\sum_{i=0}^{m} \lambda_i \nabla f_i(x_0) \right) \cdot (x - x^{(k)}) + \frac{1}{2} \left(\sum_{i=0}^{m} \lambda_i \rho_i \right) \left| \frac{x - x^{(k)}}{\sigma} \right|^2 \tag{9}$$

$$= \lambda \cdot f_i(x_0) + \sum_{j=1}^{n} h_j(x_j - x_j^{(k)}), \tag{10}$$

where

$$h_j(\delta_j) = (\lambda \cdot \nabla f(x_0)_j)\, \delta_j + \frac{1}{2\sigma_j}(\lambda \cdot \rho)\delta_j^2. \tag{11}$$

Define dual function,

$$g(\lambda) = \min_{x \in T} L(x, \lambda) \tag{12}$$

$$= \lambda \cdot f_i(x_0) + \sum_{j=1}^{n} \left(\min_{|\delta_j| \leq \sigma_j} h_j(\delta_j) \right). \tag{13}$$

Defin for each j

$$a_j = \frac{1}{2\sigma_j^2}(\lambda \cdot \rho) \tag{14}$$

$$b_j = \lambda \cdot \nabla f(x_0)_j \tag{15}$$

Note that we can write

$$b = \nabla f(x_0)^T \lambda, \tag{16}$$

where $\nabla f(x_0)$ is a matrix. We now have,

$$h_j(\delta_j) = a_j \delta_j^2 + b_j \delta_j. \tag{17}$$

The minimum of $h_j(\delta_j)$ is found at

$$\delta_j^* = -\frac{b_j}{2a_j} \text{ clamped to } [-\sigma_j, \sigma_j]. \tag{18}$$

And hence we can determine

$$g(\lambda) = \lambda \cdot f_i(x_0) + \sum_{j=1}^{n} \left(a_j \delta_j^* + b_j (\delta_j^*)^2 \right). \tag{19}$$

Now let us compute the gradient. Note that,

$$\frac{\partial a_j}{\partial \lambda_i} = \frac{\rho_i}{2\sigma_j^2}, \tag{20}$$

$$\frac{\partial b_j}{\partial \lambda_i} = \nabla f_i(x_0)_j. \tag{21}$$

If we snap to bounds, the minimum of $h_j(\lambda)$ should have gradient 0 (will have a kink, but oh well?). So let $S \subseteq \{1, \ldots, m\}$. be the indices where don't snap to bounds. Then,

$$\begin{aligned}
\frac{\partial g}{\partial \lambda_i} &= \sum_{j \in S} \left(-\frac{b_j}{2a_j} \frac{\partial b_j}{\partial \lambda_i} + \frac{b_j^2}{4a_j^2} \frac{\partial a_j}{\partial \lambda_i} \right) && (22)\\
&= \sum_{j \in S} \left(-\frac{b_j}{2a_j} \nabla f_i(x_0)_j + \frac{b_j^2}{4a_j^2} \frac{\rho_i}{2\sigma_j^2} \right). && (23)\\
\frac{\partial h_j}{\partial \lambda_i} &= \frac{b_j}{2a_j} \frac{\partial b_j}{\partial \lambda_i} - \frac{b_j^2}{4a_j^2} \frac{\partial a_j}{\partial \lambda_i} && (24)\\
&= \frac{b_j}{2a_j} \nabla f_i(x_0)_j - \frac{b_j^2}{4a_j^2} \frac{\rho_i}{2\sigma_j^2} && (25)\\
&= \delta_j \nabla f_i(x_0)_j - g_j(\lambda) \frac{\rho_i}{2\sigma_j^2}. && (26)
\end{aligned}$$

And thus,

$$\frac{\partial g}{\partial \lambda} = \nabla f(x_0) v_j + \rho \sum_{j \in S} \frac{b_j^2}{8a_j^2 \sigma_j^2}. \tag{27}$$

where v_j is a vector satisfying

$$v_j = \begin{cases} \delta_j^* & \text{if } \delta_j^* \in (-\sigma_j, \sigma_j), \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

4.2 Maximizing dual function

The dual problem is,

$$\max_{y \geq 0} g(\lambda). \tag{29}$$

We can provide g and its gradient function recursively to CCSA, which will solve it for us.

5 Main Goals

- Support sparse Jacobians
- Support affine constraints

- Does paper handle these? (First, find where paper handles box constraints.)
- Maybe think of this as a more complicated X, rather than a simple f_i.