Analytic Gradients in Trajectory Optimization

Will Whener

May 16, 2016

This document describes the method for calculating the analytic gradients for the multiple shooting approach, "rungeKutta" of the TrajOpt package. The procedure is exactly the same as that described in Russ Tedrake's Course notes [1],(Sec 12.3.3). It may look complicated, but it is essentially just a careful and repeated carrying out of the chain rule. The gradient calculation depends on the specific numerical integration approach of the dynamics and cost function. We consider a couple integration cases below: explicit Euler, and 4th order Runge-Kutta. The explicit Euler case is only considered as a simple example. The 4th order Runge-Kutta is the integration scheme used in multiple shooting approach, "rungeKutta", of the TrajOpt package.

1 Cost function gradients

Consider the scalar valued cost function,

$$J^{\boldsymbol{\alpha}} = h(t_0, t_F, \boldsymbol{x}(t_0), \boldsymbol{x}(t_F)) + \int_{t_0}^{t_F} g(t, \boldsymbol{x}(t), \boldsymbol{u}(t))dt, \tag{1}$$

for which we would like to find the gradient of $J^{\boldsymbol{\alpha}}$ with respect to some set of parameters $\boldsymbol{\alpha}$. The cost function $J^{\boldsymbol{\alpha}}$ is a function of the trajectory $(t, \boldsymbol{x}(t), \boldsymbol{u}(t))$, which is constrained by the dynamics,

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}), \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \quad t \in [t_0, t_F]. \tag{2}$$

The vector of free parameters, $\boldsymbol{\alpha}$, could represent a variety of things, for example it could be the initial condition $\boldsymbol{\alpha} = \boldsymbol{x}_0$ or it could be a control policy of the form, $\boldsymbol{u} = \boldsymbol{u}^{\boldsymbol{\alpha}}(t, \boldsymbol{x})$.

To proceed, we must first discretize the trajectory in time for purposes of numerical integration. Following the procedure of [2], we divide the trajectory into substeps evenly distributed in time and specify the state ($\boldsymbol{x}$) and control ($\boldsymbol{u}$) at each substep. Additionally, we define the value of the control at midpoints, points half way between each substep. For example, see Figure 1, where the trajectory is divided into $N = 4$ substeps: $\boldsymbol{x}[n]_{n=0,\dots,N}$, $\boldsymbol{u}[n]_{n=0,\dots,N}$, and $\boldsymbol{u}_M[n]_{n=0,\dots,N-1}$.

1.1 Explicit Euler Integration

Consider explicit Euler integration of the dynamics (2),

$$\boldsymbol{x}[n+1] = \boldsymbol{x}[n] + dt\boldsymbol{f}(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n]), \quad dt = \frac{t_F - t_0}{N}, \quad t_n = t_0 + ndt.$$

Integrating the cost function (1) in the same manner, we have

$$J^{\boldsymbol{\alpha}} = h(t_0, t_F, \boldsymbol{x}[0], \boldsymbol{x}[N]) + \sum_{n=0}^{N-1} dtg(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n]),$$

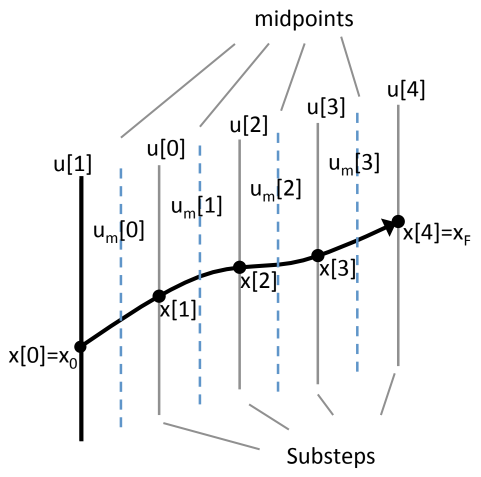

Figure 1: Time discretization of trajectory. In the particular case illustrated, there are $N = 4$ substeps. The initial time t_0 corresponds to $n = 0$ and the final time of the trajectory corresponds to $n = N$.

and the gradient is found be repeatedly carrying out the chain rule,

$$\begin{aligned}\frac{\partial J^{\boldsymbol{\alpha}}}{\partial \boldsymbol{\alpha}} =& \frac{\partial h(t_0, t_F, \boldsymbol{x}[0], \boldsymbol{x}[N])}{\partial t_0}\frac{\partial t_0}{\partial \boldsymbol{\alpha}} + \frac{\partial h(t_0, t_F, \boldsymbol{x}[0], \boldsymbol{x}[N])}{\partial t_F}\frac{\partial t_F}{\partial \boldsymbol{\alpha}} + \frac{\partial h(t_0, t_F, \boldsymbol{x}[0], \boldsymbol{x}[N])}{\partial \boldsymbol{x}[0]}\frac{\partial \boldsymbol{x}[0]}{\partial \boldsymbol{\alpha}} \\ &+ \frac{\partial h(t_0, t_F, \boldsymbol{x}[0], \boldsymbol{x}[N])}{\partial \boldsymbol{x}[N]}\frac{\partial \boldsymbol{x}[N]}{\partial \boldsymbol{\alpha}} + \frac{\partial dt}{\partial \boldsymbol{\alpha}}\sum_{n=0}^{N-1} g(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n]) \\ &+ \sum_{n=0}^{N-1} dt\left(\frac{\partial g(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])}{\partial t_n}\frac{\partial t_n}{\partial \boldsymbol{\alpha}} + \frac{\partial g(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}[n]}{\partial \boldsymbol{\alpha}} + \frac{\partial g(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}[n]}{\partial \boldsymbol{\alpha}}\right)\end{aligned}$$

The terms $\partial \boldsymbol{x}[n]/\partial \boldsymbol{\alpha}$ can be obtained while integrating the dynamics forward, we have,

$$\begin{aligned}\frac{\partial \boldsymbol{x}[n+1]}{\partial \boldsymbol{\alpha}} =& \frac{\partial \boldsymbol{x}[n]}{\partial \boldsymbol{\alpha}} + dt\left(\frac{\partial \boldsymbol{f}(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])}{\partial t}\frac{\partial t_n}{\partial \boldsymbol{\alpha}} + \frac{\partial \boldsymbol{f}(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}[n]}{\partial \boldsymbol{\alpha}} + \frac{\partial \boldsymbol{f}(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}[n]}{\partial \boldsymbol{\alpha}}\right) \\ &+ \frac{\partial dt}{\partial \boldsymbol{\alpha}}\boldsymbol{f}(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n]).\end{aligned} \tag{3}$$

and the remaining terms, $\partial \boldsymbol{x}[0]/\partial \boldsymbol{\alpha}$, $\partial \boldsymbol{u}[n]/\partial \boldsymbol{\alpha}$, $\partial t_0/\partial \boldsymbol{\alpha}$, etc. depend on the particular definition of $\boldsymbol{\alpha}$ in question. For example, take $\boldsymbol{\alpha} = \boldsymbol{x}_0$, in which case we have

$$\frac{\partial \boldsymbol{x}[0]}{\partial \boldsymbol{\alpha}} = \boldsymbol{I}, \quad \frac{\partial \boldsymbol{u}[n]}{\partial \boldsymbol{\alpha}} = 0, \quad \frac{\partial t_0}{\partial \boldsymbol{\alpha}} = 0, \quad \text{etc.}$$

Considering another possibility: $\boldsymbol{\alpha}^T = [t_0, t_F, \boldsymbol{x}_0^T, \boldsymbol{u}_0^T, (\boldsymbol{u}_0^M)^T, \boldsymbol{u}_1^T, (\boldsymbol{u}_1^M)^T ..., \boldsymbol{u}_N^T]$, i.e. an open-loop control policy, then we have,

$$\frac{\partial \boldsymbol{x}[0]}{\partial \boldsymbol{\alpha}} = \begin{bmatrix}\boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} & \dots & \boldsymbol{0}\end{bmatrix}, \quad \frac{\partial \boldsymbol{u}[0]}{\partial \boldsymbol{\alpha}} = \begin{bmatrix}\boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} & \dots & \boldsymbol{0}\end{bmatrix}, \quad \frac{\partial t_0}{\partial \boldsymbol{\alpha}} = \begin{bmatrix}1 & 0 & \dots & 0\end{bmatrix}, \quad \text{etc.}$$

Note, for the explicit Euler integration scheme, the midpoint control values were not involved, but they are involved in the 4th order Runge-Kutta scheme below.

2 Multiple-Shooting Formulation

So far we have only considered the single shooting approach. In the multiple shooting approach, the trajectory is divided into multiple segments as shown in Figure 2. Discrepancies between the endpoints of the segments are accounted for by defect constraints. If the trajectory is divided into N_S segments, this requires the introduction of the optimization parameters $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, …, $\boldsymbol{x}_{N_S-1}$. Now, the terms $\partial\boldsymbol{x}[n]/\partial\boldsymbol{\alpha}$ can be calculated separately for each shooting segment to determine gradients of the cost function and any constraint functions. For example, consider the first defect constraint of Figure 2,

$$\boldsymbol{\Delta}_0 = \boldsymbol{x}_0^+ - \boldsymbol{x}_1.$$

and take the set of optimization parameters to be

$$\boldsymbol{\alpha} = [t_0, t_F, \boldsymbol{x}_0^T, \boldsymbol{x}_1^T, \boldsymbol{x}_2^T, \boldsymbol{x}_F^T, \boldsymbol{u}_0^T, (\boldsymbol{u}_0^M)^T, \boldsymbol{u}_1^T, (\boldsymbol{u}_1^M)^T, \ldots, \boldsymbol{u}_N^T].$$

The gradient, $\partial\boldsymbol{\Delta}_0/\partial\boldsymbol{\alpha}$, requires $\partial\boldsymbol{x}_0^+/\partial\boldsymbol{\alpha}$ and $\partial\boldsymbol{x}_1/\partial\boldsymbol{\alpha}$. The first term can be obtained by carrying out the recursive equation (3), over the first shooting segment starting from

$$\frac{\partial\boldsymbol{x}[0]}{\partial\boldsymbol{\alpha}} = \frac{\partial\boldsymbol{x}_0}{\partial\boldsymbol{\alpha}} = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \boldsymbol{I} & \mathbf{0} & \ldots & \mathbf{0}\end{bmatrix}. \tag{4}$$

Similarly, to determine the gradient of second defect, the term $\partial\boldsymbol{x}_0^+/\partial\boldsymbol{\alpha}$ is evaluated using (3) over the second segment starting from

$$\frac{\partial\boldsymbol{x}[2]}{\partial\boldsymbol{\alpha}} = \frac{\partial\boldsymbol{x}_1}{\partial\boldsymbol{\alpha}} = \begin{bmatrix}\mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{I} & \ldots & \mathbf{0}\end{bmatrix}. \tag{5}$$

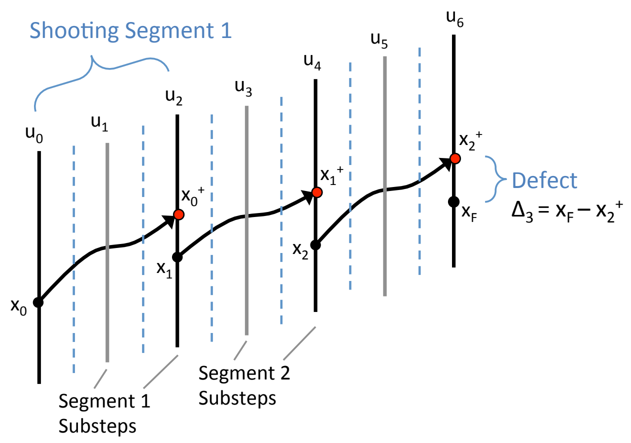

Figure 2: Multiple shooting case. The trajectory is divided into segments. In this particular case $N_S = 3$ segments and $N = 2$ substeps for each segment.

3 Other Integration Schemes

3.1 4th Order Runge-Kutta Integration

See the function `simSysGrad` inside `rungeKutta.m` of the TrajOpt package for the matlab implementation of the following equations. Consider the path cost

$$J^{\boldsymbol{\alpha}} = \int_{t_0}^{t_F} g(t, \boldsymbol{x}(t), \boldsymbol{u}(t))dt, \tag{6}$$

and employ a 4th-order Runge-Kutta scheme for integration of the dynamics and cost function,

$$J^{\boldsymbol{\alpha}} = \sum_{n=0}^{N-1} \frac{dt}{6}\left(g_0[n] + g_1[n] + g_2[n] + g_3[n]\right) \qquad \boldsymbol{x}[n+1] = \boldsymbol{x}[n] + \frac{dt}{6}\left(\boldsymbol{k}_0[n] + \boldsymbol{k}_1[n] + \boldsymbol{k}_2[n] + \boldsymbol{k}_3[n]\right)$$

$$\begin{aligned}
g_0[n] &= g(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n]) & \boldsymbol{k}_0[n] &= \boldsymbol{f}(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])\\
g_1[n] &= g(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\ \boldsymbol{k}_0, \boldsymbol{u}^M[n]) & \boldsymbol{k}_1[n] &= \boldsymbol{f}(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\ \boldsymbol{k}_0, \boldsymbol{u}^M[n])\\
g_2[n] &= g(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\ \boldsymbol{k}_1, \boldsymbol{u}^M[n]) & \boldsymbol{k}_2[n] &= \boldsymbol{f}(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\ \boldsymbol{k}_1, \boldsymbol{u}^M[n])\\
g_3[n] &= g(t_n + dt, \boldsymbol{x}[n] + dt\ \boldsymbol{k}_2, \boldsymbol{u}[n+1]) & \boldsymbol{k}_3[n] &= \boldsymbol{f}(t_n + dt, \boldsymbol{x}[n] + dt\ \boldsymbol{k}_2, \boldsymbol{u}[n+1])
\end{aligned}$$

The gradient of the cost function w.r.t. the decision parameter $\boldsymbol{\alpha}$ is

$$\frac{\partial J}{\partial \boldsymbol{\alpha}} = \sum_{n=0}^{N-1} \frac{dt}{6}\left(\frac{\partial g_0[n]}{\partial \boldsymbol{\alpha}} + \frac{\partial g_1[n]}{\partial \boldsymbol{\alpha}} + \frac{\partial g_2[n]}{\partial \boldsymbol{\alpha}} + \frac{\partial g_3[n]}{\partial \boldsymbol{\alpha}}\right) + \frac{1}{6}\frac{\partial dt}{\partial \boldsymbol{\alpha}}\left(g_0[n] + g_1[n] + g_2[n] + g_3[n]\right)$$

where

$$\frac{\partial \boldsymbol{x}[n+1]}{\partial \boldsymbol{\alpha}} = \frac{\partial \boldsymbol{x}[n]}{\partial \boldsymbol{\alpha}} + \frac{dt}{6}\left(\frac{\partial \boldsymbol{k}_0[n]}{\partial \boldsymbol{\alpha}} + \frac{\partial \boldsymbol{k}_1[n]}{\partial \boldsymbol{\alpha}} + \frac{\partial \boldsymbol{k}_2[n]}{\partial \boldsymbol{\alpha}} + \frac{\partial \boldsymbol{k}_3[n]}{\partial \boldsymbol{\alpha}}\right) + \frac{1}{6}\frac{\partial dt}{\partial \boldsymbol{\alpha}}\left(\boldsymbol{k}_0[n] + \boldsymbol{k}_1[n] + \boldsymbol{k}_2[n] + \boldsymbol{k}_3[n]\right)$$

Now, all that's left is to obtain the terms, $\partial \boldsymbol{k}_i[n]/\partial \boldsymbol{\alpha}$ and $\partial g_i[n]/\partial \boldsymbol{\alpha}$. This is where things start to get a bit ugly, but, again, it's just a repeated carrying out of the chain rule,

$$\begin{aligned}
\frac{\partial \boldsymbol{k}_0}{\partial \boldsymbol{\alpha}} =& \frac{\partial \boldsymbol{f}(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])}{\partial t}\frac{\partial t_n}{\partial \boldsymbol{\alpha}} + \frac{\partial \boldsymbol{f}(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}[n]}{\partial \boldsymbol{\alpha}} + \frac{\partial \boldsymbol{f}(t_n, \boldsymbol{x}[n], \boldsymbol{u}[n])}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}[n]}{\partial \boldsymbol{\alpha}},\\
\frac{\partial \boldsymbol{k}_1}{\partial \boldsymbol{\alpha}} =& \frac{\partial \boldsymbol{f}(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\boldsymbol{k}_0, \boldsymbol{u}^M[n])}{\partial t}\left(\frac{\partial t_n}{\partial \boldsymbol{\alpha}} + 0.5\frac{\partial dt}{\partial \boldsymbol{\alpha}}\right) +\\
& \frac{\partial \boldsymbol{f}(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\boldsymbol{k}_0, \boldsymbol{u}^M[n])}{\partial \boldsymbol{x}}\left(\frac{\partial \boldsymbol{x}[n]}{\partial \boldsymbol{\alpha}} + 0.5\boldsymbol{k}_0\frac{\partial dt}{\partial \boldsymbol{\alpha}} + 0.5dt\frac{\partial \boldsymbol{k}_0}{\partial \boldsymbol{\alpha}}\right) +\\
& \frac{\partial \boldsymbol{f}(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\boldsymbol{k}_0, \boldsymbol{u}^M[n])}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}^M[n]}{\partial \boldsymbol{\alpha}},\\
\frac{\partial \boldsymbol{k}_2}{\partial \boldsymbol{\alpha}} =& \frac{\partial \boldsymbol{f}(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\boldsymbol{k}_1, \boldsymbol{u}^M[n])}{\partial t}\left(\frac{\partial t_n}{\partial \boldsymbol{\alpha}} + 0.5\frac{\partial dt}{\partial \boldsymbol{\alpha}}\right) +\\
& \frac{\partial \boldsymbol{f}(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\boldsymbol{k}_1, \boldsymbol{u}^M[n])}{\partial \boldsymbol{x}}\left(\frac{\partial \boldsymbol{x}[n]}{\partial \boldsymbol{\alpha}} + 0.5\boldsymbol{k}_1\frac{\partial dt}{\partial \boldsymbol{\alpha}} + 0.5dt\frac{\partial \boldsymbol{k}_1}{\partial \boldsymbol{\alpha}}\right) +\\
& \frac{\partial \boldsymbol{f}(t_n + 0.5dt, \boldsymbol{x}[n] + 0.5dt\boldsymbol{k}_1, \boldsymbol{u}^M[n])}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}^M[n]}{\partial \boldsymbol{\alpha}},\\
\frac{\partial \boldsymbol{k}_3}{\partial \boldsymbol{\alpha}} =& \frac{\partial \boldsymbol{f}(t_n + dt, \boldsymbol{x}[n] + dt\boldsymbol{k}_2, \boldsymbol{u}[n+1])}{\partial t}\left(\frac{\partial t_n}{\partial \boldsymbol{\alpha}} + \frac{\partial dt}{\partial \boldsymbol{\alpha}}\right) +\\
& \frac{\partial \boldsymbol{f}(t_n + dt, \boldsymbol{x}[n] + dt\boldsymbol{k}_2, \boldsymbol{u}[n+1])}{\partial \boldsymbol{x}}\left(\frac{\partial \boldsymbol{x}[n]}{\partial \boldsymbol{\alpha}} + \boldsymbol{k}_2\frac{\partial dt}{\partial \boldsymbol{\alpha}} + dt\frac{\partial \boldsymbol{k}_2}{\partial \boldsymbol{\alpha}}\right) +\\
& \frac{\partial \boldsymbol{f}(t_n + dt, \boldsymbol{x}[n] + dt\boldsymbol{k}_2, \boldsymbol{u}[n+1])}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}[n+1]}{\partial \boldsymbol{\alpha}},
\end{aligned}$$

The terms $\partial g_i[n]/\partial \boldsymbol{\alpha}$ are very similar to $\partial \boldsymbol{k}_i[n]/\partial \boldsymbol{\alpha}$. In each instance, simply replace $\boldsymbol{k}_i$ with g_i and $\boldsymbol{f}$ with g to obtain $\partial g_i[n]/\partial \boldsymbol{\alpha}$.

References

[1] R. Tedrake, *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832)*. Downloaded on May 9, 2015 from http://underactuated.mit.edu/, 2015. [Online]. Available: http://underactuated.csail.mit.edu/underactuated.html

[2] J. Betts, *Practical Methods for Optimal Control Using Nonlinear Programming*, ser. Advances in design and control. Society for Industrial and Applied Mathematics, 2001. [Online]. Available: https://books.google.com/books?id=Yn53JcYAeaoC