Collocation Methods in General and Conditioning of the Corresponding Karush-Kuhn-Tucker Systems in Particular

Fredrik Magnusson

Department of Automatic Control
Faculty of Engineering
Lund University, Sweden

February 21, 2014

Outline

Outline

1. Dynamic optimization
2. Conditioning of nonlinear programs

JModelica.org

Dynamic optimization

Dynamic optimization problems involve

- a dynamic system
- a continuous-time model of the system dynamics
- constraints enforcing the model equations
- a time horizon, which may be infinite or finite, free or fixed

Applications include

- optimal control, either open-loop or e.g. model predictive control
- grey-box identification
- state estimation (typically moving horizon estimation)

In our course in Nonlinear Control, we study optimal control problems of the form

$$
\begin{array}{ll}
\text{minimize} & \phi(t_f, x(t_f)) + \int_0^{t_f} L(x(t), u(t))\,\mathrm{d}t, \\
\text{with respect to} & t_f, x, u, \\
\text{subject to} & \dot{x} = f(x(t), u(t)), \\
& x(0) = x_0, \\
& g_e(t, u(t)) = 0, \\
& g_i(t, u(t)) \leq 0, \\
& \psi(x(t_f)) = 0, \\
& \forall t \in [0, t_f].
\end{array}
$$

My work treats various generalizations of the above problem, most notably handling differential-algebraic equation systems instead of ordinary differential equations and state constraints.

Pontryagin's maximum principle

- Using Pontryagin's maximum principle, this results in a boundary value problem.
- Not feasible to solve analytically in many practical applications
- One possible approach is to solve the boundary value problem numerically. This is called an indirect approach, and was the state of the art until the 1970s.
- Two major weaknesses
  - the switching structure of the inequality constraints can be difficult to find
  - sensitive to initial guesses of adjoint states

Direct approach

Another approach is to discretize the problem before establishing optimality conditions.

- Results in a mathematical program
- Optimality conditions given by the Karush–Kuhn–Tucker (KKT) conditions
- This is called a direct approach, and is most commonly used today.
- Two major categories of direct approaches: collocation and multiple shooting

Collocation methods

Main idea is to approximate system trajectories by polynomials:

- Divide the time horizon $[t_0, t_f]$ into a finite number of elements
- Approximate the time-variant variables in each element by a polynomial
- Force this polynomial to satisfy all the constraints in n_c points
- This uniquely determines a polynomial of degree $n_c - 1$ by interpolation

Nonlinear program

- The result is a nonlinear program on the form

$$\begin{aligned} &\text{minimize} && f(x), \\ &\text{with respect to} && x \in \mathbb{R}^n, \\ &\text{subject to} && x_L \leq x \leq x_U, \\ &&& g(x) = 0, \\ &&& h(x) \leq 0. \end{aligned}$$

- NLP solution approximates the solution to the original dynamic optimization problem.
- JModelica.org uses IPOPT to solve NLPs.

Outline

Convex programming

In our course in Convex Optimization, we study numerical solution of convex programs of the form

$$
\begin{aligned}
\text{minimize} \quad & f(x), \\
\text{with respect to} \quad & x \in \mathbb{R}^n, \\
\text{subject to} \quad & Ax = b, \\
& h(x) \leq 0,
\end{aligned}
$$

where f and h are convex. The problem is solved using the barrier method and using Newton's method for the inner iterations.

KKT system

The main step of Newton's method is solving the KKT system (linearization of modified KKT conditions)

$$\begin{bmatrix} t\nabla^2 f(x) + \nabla^2\phi(x) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \nu \end{bmatrix} = - \begin{bmatrix} t\nabla f(x) + \nabla \phi(x) \\ 0 \end{bmatrix},$$

where t is the (inverse) barrier parameter, ϕ is the barrier term, Δx is the primal step and $\Delta \nu$ is (or rather, would be) the dual step.

Problem scaling

The scaling/conditioning of the problem becomes crucial for the performance of the method (especially for large-scale problems). John Betts gives the following “hints” for achieving well-scaledness of an NLP:

- normalize the decision variables to have unitary magnitude
- normalize the constraint residuals and objective to have unitary magnitude
- normalize the rows and columns of the Jacobian to have unitary magnitude
- normalize the constraint residuals so that the dual variables have unitary magnitude
- somehow achieve unitary condition number of projected Hessian
- somehow achieve unitary condition number of the KKT matrix

Problem scaling

The scaling/conditioning of the problem becomes crucial for the performance of the method (especially for large-scale problems). John Betts gives the following “hints” for achieving well-scaledness of an NLP:

- normalize the decision variables to have unitary magnitude
- normalize the constraint residuals and objective to have unitary magnitude
- normalize the rows and columns of the Jacobian to have unitary magnitude
- normalize the constraint residuals so that the dual variables have unitary magnitude
- somehow achieve unitary condition number of projected Hessian
- somehow achieve unitary condition number of the KKT matrix

KKT condition number as a performance measure

- Ideas for automating residual scaling and improving constraint Jacobian scaling
- How to evaluate these ideas? Execution time and number of iterations unreliable!
- Idea: use KKT condition number as a performance measure

Condition number

- Condition number κ of matrix A defined as

$$\kappa(A) := \frac{\sigma_{\text{max}}(A)}{\sigma_{\text{min}}(A)}$$

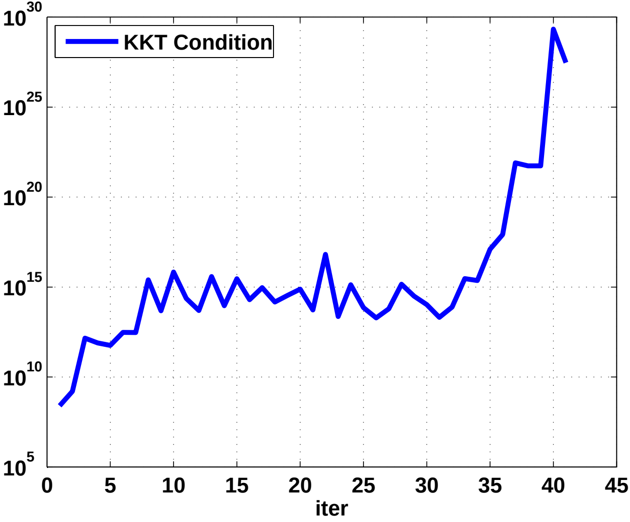
- Interesting when solving linear systems $Ax = b$, since e.g.

$$\frac{||\delta x||}{||x||} \leq \kappa(A) \frac{||\delta b||}{||b||}.$$

CSTR case study

- Extract the different parts of the KKT matrix for each iteration in IPOPT.
- A simple case study of how the KKT condition number varies in each iteration during the solution of a simple optimal control problem
- Temperature control of a two-state continuous stirred-tank reactor (CSTR) (small but highly nonlinear)

KKT condition number for CSTR



IPOPT KKT matrix

IPOPT first transforms the NLP to the form

$$\begin{aligned} \text{minimize} \quad & f(x), \\ \text{subject to} \quad & x^L \leq x \leq x^U, \\ & g(x) = 0, \end{aligned}$$

and then works with the KKT system

$$\begin{bmatrix} W + \Sigma + \delta_W I & \nabla g^T(x) \\ \nabla g(x) & -\delta_g I \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \nu \end{bmatrix} = - \begin{bmatrix} \nabla f(x) + \mu \nabla \phi(x) + \nabla g^T \nu \\ g(x) \end{bmatrix},$$

where W is the Hessian of the Lagrangian, δ_W and δ_g are regularization parameters and Σ is a diagonal matrix defined by

$$\Sigma_{i,i} := \frac{z_i^L}{x_i - x_i^L} + \frac{z_i^U}{x_i^U - x_i},$$

where z^L and z^U are dual variables for the lower and upper bounds on x, respectively.

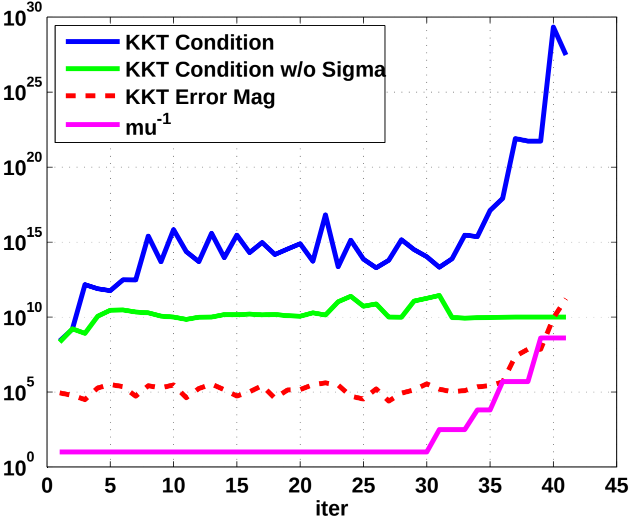
Benign ill-conditioning

- The inherent ill-conditioning of interior-point methods has been known for a long time
- But it was not understood until the late 90s
- The ill-conditioning is in large part caused by Σ (and μ)

So, how to measure the effects of problem scaling if the condition number of the KKT matrix is insignificant? Two unfounded ideas:

1. Condition number of KKT matrix without the Σ term
2. $||K \backslash K\mathbf{1} - \mathbf{1}||_{\text{RMS}}\epsilon_{\text{mach}}^{-1}$

KKT condition number for CSTR revisited



Conclusions

- Problem conditioning is important (although I have not really demonstrated that), and only partly automated
- Problem conditioning is hard
- Evaluating various methods for improving problem conditioning was a lot more difficult than I had initially hoped for. Hopefully to be continued...

The end

Thank you for listening!

The End