Contents

Dual problems and KKT

In this file, we solve a simple convex quadratic program, and verify that the KKT conditions hold at the optimium (solvwed by `quadprag`). We then separately formulate the dual problem (also a convex quadratic program), solve it, and verify that there is no duality gap (ie., the maximum of the dual equals the minimum of the primal).

```
format compact
format short e
```

Description of problem

The problem is taken from the introductory slides on optimization. A point $\bar{x} \in \mathbf{R}^n$ is given, and the problem is

$$\min_x \|x - \bar{x}\|^2 \quad \text{s.t.} \quad -1 \leq x \leq 1$$

This is a **convex quadratic program** as it involves a convex quadratic objective function along with linear inequality constraints. We refer to it as the *primal* problem (to differentiate it from the *dual* problem which will introduce later). Rewrite in the notation of `quadprog`, namely

$$\min_x f(x) \quad \text{s.t.} \ Ax \leq b$$

where

$$f(x) = (x - \bar{x})^T (x - \bar{x}) = \frac{1}{2} x^T (2 I_2) x - 2 \bar{x}^T x + \bar{x}^T \bar{x}$$

and

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

The `quadprog` function minimizes functions of the form

$$\frac{1}{2}x^T H x + w^T x$$

for a given symmetric, positive-semidefinite matrix H and columen vector w. Translating to our problem yields

$$H = 2I_2, \qquad w = -2\bar{x}$$

Note that `quadprog` does not include a constant-term, so the user (you!) need to keep track of the constant term on your own, although it has no effect on the value of the optimal x (or the optimal dual variables).

In the slides, the problem is 2-dimensional ($n = 2$), and the specific value for the point $\bar{x}$ is given.

```
nx = 2;
xbar = [3;2];
H = 2*eye(nx);
w = -2*xbar;
A = [eye(nx);-eye(nx)];
b = ones(2*nx,1);
c0 = xbar'*xbar;
```

Solution using `quadprog`

```
[xOpt,objOpt,EXITFLAG,OUTPUT,LAMBDA] = quadprog(H,w,A,b);
primalF = objOpt + c0;
```

```
Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the optimality tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.
```

`EXITFLAG` gives status of the optimization. A value of `1` indicates success, and that a point satisfying the optimiality conditions has been found.

```
disp(EXITFLAG)
```

```
     1
```

`OUTPUT` is a stucture with information about the iteration that took place in finding the solution. You can look at it and the documentation for more information.

```
OUTPUT
```

```
OUTPUT =
          message: 'Minimum found that satisfies the constraints....'
        algorithm: 'interior-point-convex'
    firstorderopt: 4.1655e-09
  constrviolation: 0
       iterations: 4
     cgiterations: []
```

LAMBDA is a structure containing the optimal Lagrange multipliers. The function quadprog allows for 3 types of constraints

- general inequality, of the form $Ax \leq b$, which we used;
- general equality, of the form $A_e x = b_e$, which we **did not** use;
- "box" constraints of the form $L \leq x \leq U$, which we **did not** use (but could have, since the constraints in this problem are nothing more than simple box constraints), and hence are interpreted as having set $L = -\inf, U = \inf$.

The fields in LAMBDA are the multipliers associated with those constraints. We expect:

- a 4-by-1 vector of nonnegative values for the multipliers associated with the inequality constraints (the u vector in the lecture notes);
- an 0-by-1 (empty) for the multipliers associated with the equality constraints (the v vector in the lecture notes);
- two, 2-by-1 zero-vectors for the inactive box inequality constraints

Indeed, this is the structure of LAMBDA

```
LAMBDA
```

```
LAMBDA =
    ineqlin: [4x1 double]
      eqlin: [0x1 double]
      lower: [2x1 double]
      upper: [2x1 double]
```

For reference below, assign the multipliers associated with the inequality constraints to a variable uOpt

```
uOpt = LAMBDA.ineqlin;
```

In the code below, we will verify that the values of the multipliers certify that xOpt is a local optimum, and since the problem is convex, it further means that xOpt is a global optimum.

General form of optimization problem

The general form is

$$\min_x f(x) \ \text{ s.t. } g(x) \leq 0$$

where f is a scalar-valued objective, and g is a vector-valued function representing all of the inequality constraints. In the next code section, we carefully express f and g for this problem, along with their gradients. Recall that gradients play a key role in optimality conditions (simple results in calculus for unconstrained problems in one variable, all the way to the KKT conditions for constrained problems in many variables).

(f, g) and their gradients

In order to verify that the entries in `X` and `LAMBDA` satisfy the KKT conditions, we need to calculate the gradient of f and g . Simple calculation reveals

$$f(x) = (x - \bar{x})^T (x - \bar{x}) \quad \Rightarrow \quad \nabla f = 2(x - \bar{x})^T$$

and

$$g(x) = \begin{bmatrix} x_1 - 1 \\ x_2 - 1 \\ -x_1 - 1 \\ -x_2 - 1 \end{bmatrix} \quad \Rightarrow \quad \nabla g = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}$$

In general, of course, we have

$$g(x) = Ax - b \quad \Rightarrow \quad \nabla g = A$$

Make function handles for both of these, for clarity in the code cells below.

```
gradf = @(x) 2*(x-xbar)';
gradg = @(x) A;
g = @(x) A*x-b;
```

Check KKT optimality conditions: #1

Let x^* and u^* denote the optimal values for the decision variable and the multipliers associated with the inequality constraints. One of the KKT conditions for optimality is

$$0 = \nabla f(x^*) + u^{*T} \nabla g(x^*)$$

We can easily make this calculation with al of the variables defined so far.

```
gradf(xOpt) + uOpt'*gradg(xOpt)
```

```
ans =
  -8.8818e-16  -8.8818e-16
```

Task: Examine each term by hand, and make sure everything is as you would expect.

Check KKT optimality conditions: #2

Two other conditions for optimality are

$$0 \leq u_i^*, \quad g_i(x^*) \leq 0, i = 1, ...m$$

Verify these, first looking at `uOpt`

```
uOpt
```

```
uOpt =
   4.0000e+00
   2.0000e+00
   2.0828e-09
   9.0496e-10
```

and then the value g at `xOpt`

```
g(xOpt)
```

```
ans =
  -3.0006e-11
  -1.2793e-09
  -2.0000e+00
  -2.0000e+00
```

Check KKT optimality conditions: #3

Finally, it must be that the individual components of u and $g(x^*)$ are related, in the the elementwise product should be identically 0.

$$0 = u_i^* g_i(z^*), i = 1, ...m$$

```
uOpt.*g(xOpt)
```

```
ans =
  -1.2002e-10
  -2.5586e-09
  -4.1655e-09
  -1.8099e-09
```

Dual

The Dual problem will be derived and explained by Prof. Packard. It's also a constrained convex quadratic problem. Below we solve it and indeed the cost is a lower bound to the original (Primal) cost. In fact, since **strong duality** holds for essentially all convex quadratic programs, the optimal objectives of the primal and dual are equal.

```
Hd = -A*inv(H)*A';
wd = -(A*inv(H)*w + b);
c0d = c0 - w'*inv(H)*w/2;
```

```
lb = zeros(4,1);
[xDual,objDual] = quadprog(-Hd,-wd,[],[],[],[],lb,[]);
dualF = -objDual + c0d;
disp(['  Optimal Dual Objective: ' num2str(dualF)]);
disp(['Optimal Primal Objective: ' num2str(primalF)]);
```

```
Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the optimality tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.



  Optimal Dual Objective: 5
Optimal Primal Objective: 5
```

Generalizing to any convex quadratic program

If you are interested, you can generalize this script to check the KKT conditions (on the variables returned by `quadprog`) for any convex quadratic program.

Attribution

ME C231A and EECS C220B, UC Berkeley, Fall 2016

Published with MATLAB® R2016a