Progress Report until July 4th

Zichao Di

July 11, 2011

1 Linear and Bilinear Control problem

- The original linear control problem has the form $\min \|y - y\|^2 + \frac{\mu}{2}\|u\|^2 = f(u)$ where $y = -A^{-1}(u + f)$
- The first exprement I tried is to apply the same idea of modifing the coarse grid bound constraint as in 1d-control so that no variable will go infeasible, but this attempt failed since
  - on the coarse grid, the corresponding constraint will be too limited so that variables are barely moving, then the coarse grid relaxation won't give a descent direction to fine level.
  - The second reason is this idea is not general to MG/OPT and will be very complex to the higher dimension.
- Also, during the process, notice that the search direction is always 10 times tinier than the step to solution,and this may imply that the scaling of original problem is an issue.
  - The first try is to scale the original problem by h^2, it turns out that MG/OPT for unconstrained even failed, the reason is according to the shifted problem, our goal should be to have $\nabla f_H(x_H) \approx I_h^H \nabla f_h(x_h)$. If we use $\|u\|^2 = h^2 u_h^T u_h$, then the gradients will be $\mu h^2 u_h$ and $\mu H^2 u_H$. If we use $\|u\|^2 = u_h^T u_h$, then the gradients will beμu_h and μu_H, so $\nabla f_H(x_H) \approx I_h^H \nabla f_h(x_h)$.
  - Motivated by the fact that the search direction is tiny, so I am trying enlarge the direction by multiplying the update matrix by 2, in order to keep the relation $I_H^h = c^2 (I_h^H)^T$, in the mean time, divide the downdate matrix by 2. And the improved result is shown in figure 1. i.e. Let $A = \begin{bmatrix} \dots & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & \dots & \frac{1}{2} & 1 & \frac{1}{2} & \dots & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & \dots \end{bmatrix}$ $I_H^h = 2A^T$; $I_h^H = \frac{1}{8}A$. Traditonal setting: $I_H^h = A^T$; $I_h^H = \frac{1}{4}A$
  - Apply the same idea of modified update and downdate matrix to the bilinear control problem by MG/OPT-TN, I got a good result too as shown in figure 2.

– Next thing I tried is to solve the same problem by fmincon first with traditional transform operators: the output is:

```
options = optimset('Algorithm','interior-point','Display','iter', 'GradObj','on
 'Hessian','lbfgs','MaxIter',nit, 'TolFun', 1e-8, 'InitBarrierParam',1,
'AlwaysHonorConstraints','none','InitTrustRegionRadius',63);

mesh grid is 15
J = 2.389319077691188e+006
                                                First-order      Norm of
 Iter F-count            f(x)  Feasibility   optimality         step
    0       1   2.389319e+006   0.000e+000   1.620e-002
J = 2.389319134502814e+006
    1       2   2.389319e+006   0.000e+000   2.264e-001   6.083e-002
J = 2.389319078439562e+006
    2       3   2.389319e+006   0.000e+000   2.203e-002   5.523e-002
J = 2.389319077362157e+006
    3       4   2.389319e+006   0.000e+000   1.163e-002   2.281e-003
J = 2.389319077565342e+006
J = 2.389319077565342e+006
    4       6   2.389319e+006   0.000e+000   8.475e-003   1.056e-003
J = 2.389319077470120e+006
```

2 1-D Laplacian Problem

- In this problem, the best result for the fine level gradient evaluation is obtained from the modified bound constraint which different grid has its corresponding bound constraint.
- I also tried the same setting up of constraint using fmincon as underyling relaxation, the result is shown in figure 3, and I think the reason for fmincon working well is also that the new constraint will always keep variables feasible.
- The next thing I tried is using bending line search in MG/OPT with original constriant. The result is shown in figure 4. The reason for the stalling of MG/OPT is when the variable is close to solution, manually fix the infeasible point at the bounds won't decrease the merit function so that the step length is either zero or very tiny.

3 Next Step:

Try to find the right scale between $I_h^H \nabla f_h(v_h)$ and $\nabla f_H(I_h^H v_h)$ without changing the traditional transform operator.

4 Interior-Point Algorithm

The interior-point algorithm uses these options:

AlwaysHonorConstraint: The default 'bounds' ensures that bound constraints are satisfied at every iteration. Disable by setting to 'none'.

HessFcn: Function handle to a user-supplied Hessian (see Hessian). This is used when the Hessian option is set to 'user-supplied'.

Hessian: Chooses how fmincon calculates the Hessian (see Hessian). The choices are:
* 'bfgs' (default)
* 'fin-diff-grads'
* 'lbfgs'
* 'lbfgs',Positive Integer
* 'user-supplied'

HessMult: Handle to a user-supplied function that gives a Hessian-times-vector product (see Hessian). This is used when the Hessian option is set to 'user-supplied'.

InitBarrierParam: Initial barrier value, a positive scalar. Sometimes it might help to try a value above the default 0.1, especially if the objective or constraint functions are large.

InitTrustRegionRadius : Initial radius of the trust region, a positive scalar. On badly scaled problems it might help to choose a value smaller than the default $\sqrt{n}$, where n is the number of variables.

MaxProjCGIter : A tolerance (stopping criterion) for the number of projected conjugate gradient iterations; this is an inner iteration, not the number of iterations of the algorithm. This positive integer has a default value of 2*(numberOfVariables - numberOfEqualities).

ObjectiveLimit: A tolerance (stopping criterion) that is a scalar. If the objective function value goes below ObjectiveLimit and the iterate is feasible, the iterations halt, since the problem is presumably unbounded. The default value is -1e20.

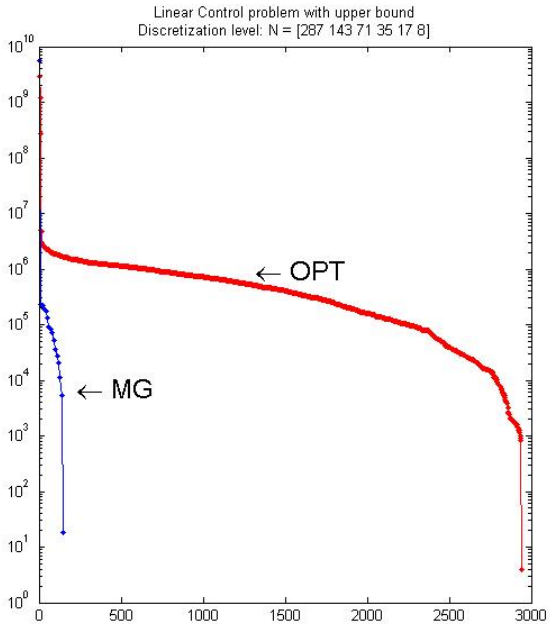
ScaleProblem: The default 'obj-and-constr' causes the algorithm to normalize all constraints and the objective function. Disable by setting to 'none'.

SubproblemAlgorithm: Determines how the iteration step is calculated. The default, 'ldl-factorization', is usually faster than 'cg' (conjugate gradient),

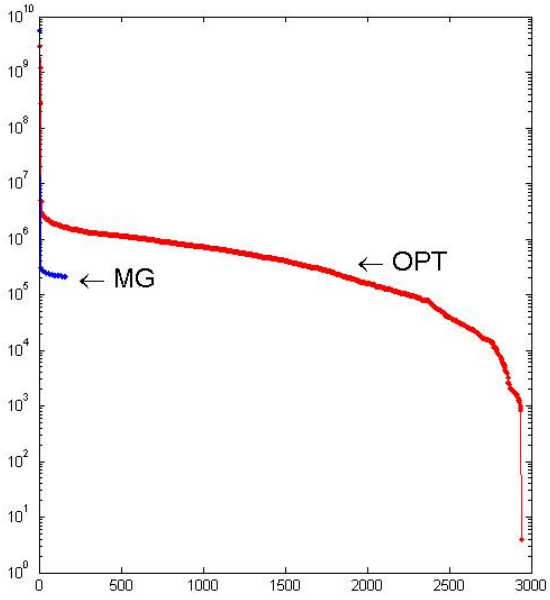
though 'cg' might be faster for large problems with dense Hessians

TolProjCG: A relative tolerance (stopping criterion) for projected conjugate gradient algorithm; this is for an inner iteration, not the algorithm iteration. This positive scalar has a default of 0.01.

TolProjCGAbs: Absolute tolerance (stopping criterion) for projected conjugate gradient algorithm; this is for an inner iteration, not the algorithm iteration. This positive scalar has a default of 1e-10.



(a) $I_H^h = 16(I_h^H)^T$



(b) $I_H^h = 4(I_h^H)^T$

Figure 1: Linear control with upper bound by MG/OPT-TN

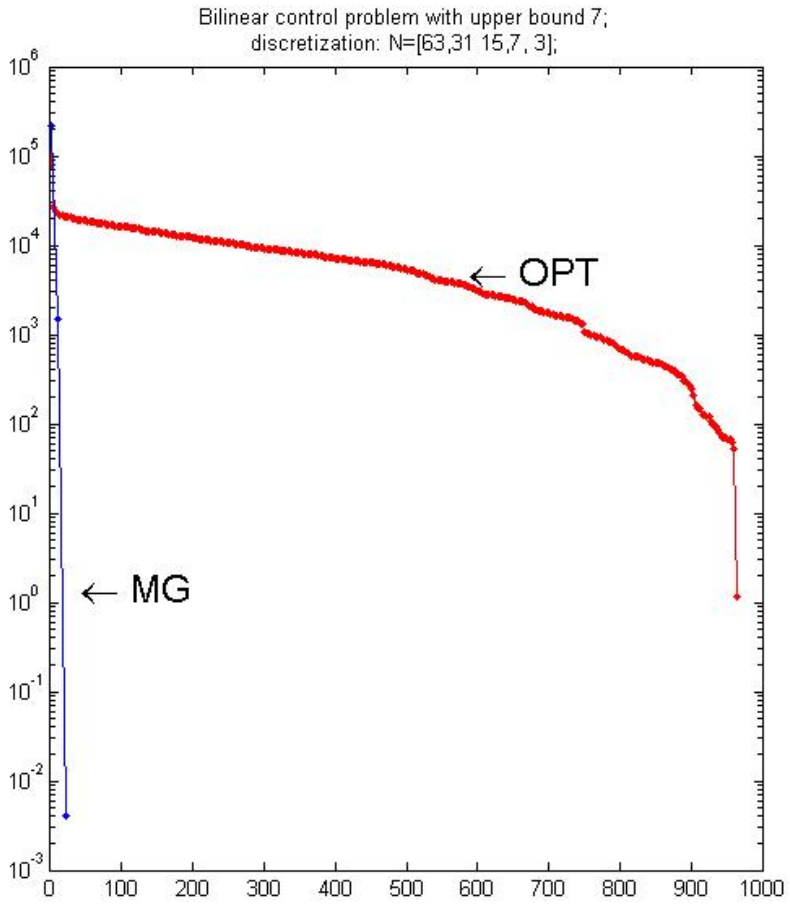


Figure 2: Bilinear control problem with upper bound by MG/OPT-TN

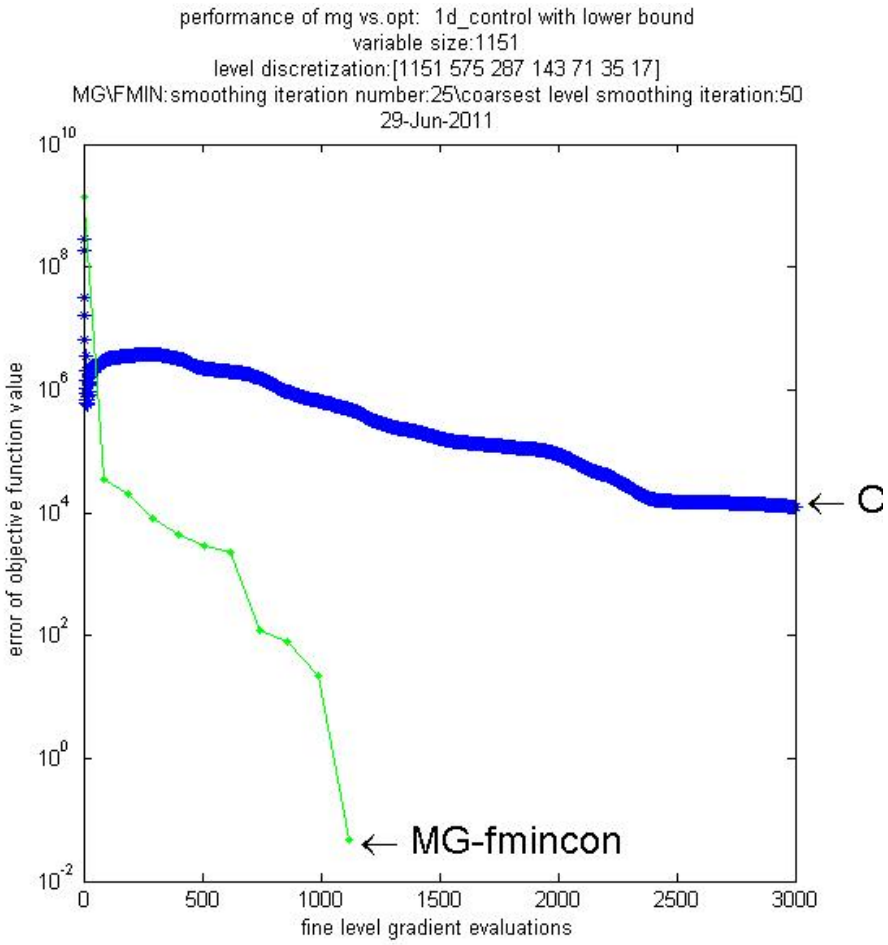


Figure 3: 1-D control problem with upper bound using fmincon

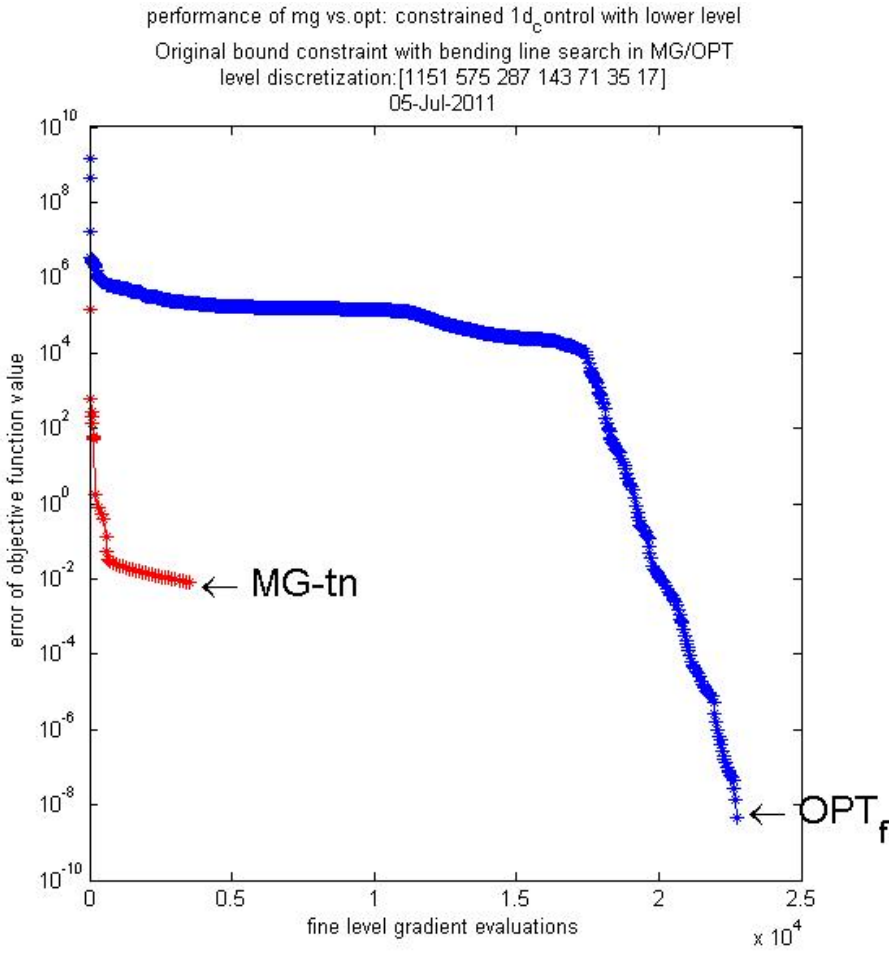


Figure 4: 1-D control problem with lower bound using bending line search with normal constraint