

Instructor: Michael Saunders Spring 2009

Project Due Thursday June 11

GAMS and Basis Pursuit

GAMS [5] is one of the widely used algebraic modeling languages for formulating optimization problems of many types. Our class on large-scale numerical optimization would be incomplete without giving you some experience with running a GAMS model. Luckily GAMS is available on the campus Linux servers, and we want to keep it that way by actually using it(!). We could say the same about AMPL [1].

The project will also help you learn about Basis Pursuit (BP) and Basis Pursuit Denoising (BPDN) [3], which helped popularize the concept of finding sparse solutions to under-determined systems (also known as ℓ^1 -minimization and *sparse recovery*).¹

Given a signal $b \in R^m$, signal analysis seeks a linear combination of certain waveforms that reconstruct b . The waveforms may be sinusoids (the traditional Fourier representation) or other “dictionaries” such as wavelets, chirplets, warplets, curvelets, etc. Many of the dictionaries are already “overcomplete” in the sense of containing more than m waveforms. Combining one or more dictionaries gives a matrix $A \in R^{m \times n}$ with $m < n$. Basis Pursuit solves the ℓ^1 problem

$$\min_x \|x\|_1 \quad \text{subject to} \quad Ax = b \quad (\text{BP})$$

to obtain a solution x that is likely to be *sparse*.² Problem (BP) is equivalent to the linear program

$$\min_{v,w} e^T(v+w) \quad \text{subject to} \quad Av - Aw = b, \quad v, w \geq 0, \quad (\text{BPLP})$$

where e is a vector of 1s and $x = v - w$. From linear programming theory we know that a solution exists with at most m nonzero v_j or w_j , and that at most one of each pair (v_j, w_j) will be nonzero for all $j = 1$ to n . If the columns of A contain “good” basis functions for b , almost all elements of v and w will be zero.

Since signals are likely to be noisy, a more realistic model is the BP denoising problem

$$\min_{x,r} \lambda \|x\|_1 + \frac{1}{2} r^T r \quad \text{subject to} \quad Ax + r = b, \quad (\text{BPDN})$$

in which the 1-norm of x is balanced against the 2-norm (squared) of a residual vector r . The size of the scalar parameter λ greatly affects the sparsity of x .

¹In turn, BP has led to *compressed sensing* [2, 4] and such concepts as the *exact reconstruction principle* (ERP), the *compressible reconstruction principle* (CRP), and the *uniform uncertainty principle* (UUP) [6].

²If the objective were $\min \|x\|_p$ with $p \rightarrow 0^+$, the solution would be as sparse as possible. However, the problem would no longer be convex and its solution would be *much* harder to find; in fact NP-hard. The choice $p = 1$ gives the closest convex approximation to the ℓ^0 problem.

The GAMS model `ajax.gms`

The GAMS Model Library <http://www.gams.com/modlib/modlib.htm> contains many types of optimization problem (LP, NLP, MIP, MINLP, ...) arising from many application areas (agricultural economics, chemical engineering, finance, ...). The Management Science and OR section contains a small linear programming model named “ajax”. It models a paper manufacturer wishing to find a production plan that maximizes monthly profits as the company produces four grades of paper on three machines, given a fixed machine availability (hours per month) and a fixed demand (tons per month). Also known are the machine production rates (tons per hour) and production costs and selling price (dollars per ton) for each kind of paper.

The variables are a 2D array `outp(g,m)` of nonnegative outputs (tons per month of each grade of paper by each machine).

File `ajaxBP.gms`

We have taken the `ajax` model as a simple but realistic example of a linear program

$$\min_x c^T x \quad \text{subject to} \quad Ax = b, \quad x \geq 0. \quad (\text{LP})$$

Our aim is to learn some parts of the GAMS language while experimenting with a Basis Pursuit version of the LP model. Rather than writing new GAMS statements (nontrivial without a manual!), you will simply study and run the modified GAMS model and conduct some “good guess” reverse engineering.

Download file `ajaxBP.gms` from <http://www.stanford.edu/class/msande318/> by clicking on `Homework` and then on the filename.

Project tasks

1. Write down the BP problem associated with problem (LP) above. In other words, which LP problem is likely to give a sparse feasible solution for (LP)? Note that we ignore the $c^T x$ objective, but the bounds $x \geq 0$ are part of problem (LP). The required BP problem doesn't need to be as big as problem (BPLP).
2. The `Sets` statement defines two sets `m` and `g`, giving names to each machine and each grade of paper. What are the names and dimensions of four associated data matrices? (They are defined with varying syntax.)
3. Note the `Variables` and `Positive Variable` statements. In your judgement, which upper and lower bounds do the variables have?
4. Note the `Equations` statement. The original linear objective in the `ajax` model is defined by a certain linear equation. What is the name of that equation, and what is the equation itself?
5. Suppose we know that `profit` is going to be nonnegative. We might try to help the optimizer by giving `profit` a lower bound of zero. Can you give a reason why this would not be a good idea?

6. Note the `Models` statement. Write down algebraically the linear constraints and bounds in the original `ajax` LP model.
7. We have added two new equations `BP` and `BPDN` to define the objective function for each of two new models. Write these objective functions algebraically.
8. Note that `sqr(r)` is a built-in function for specifying r^2 . We could have written `r**2`, but GAMS would evaluate this as `exp[2*log(r)]`. Why would this be a danger?
9. GAMS allows us to write `power(r,n)` if `n` is an integer (positive or negative). Can you suggest why `sqr(r)` is better if we know that `n = 2`? (Remember that GAMS is generating gradients for us behind the scenes.)
10. Transfer `ajaxBP.gms` to your account on the campus file system. (If you wish, you can copy it from `/afs/ir.stanford.edu/class/msande318/www/homework/ajaxBP.gms`.) Then logon to one of the `bramble` machines (64-bit 4-core Intel Xeon 3.00GHz Dell PowerEdge 1850 servers running Ubuntu 9.04) and run our three models by typing `gams ajaxBP` at the command line.

The output file will be called `ajaxBP.lst`.

Note that each `Solve` statement specifies a model, a problem type, and an objective function. Which solvers did GAMS use by default for each model? How many iterations did each solve require?

11. Typically the solvers output logfiles as they iterate. You can tell GAMS to include the logfile in its `.lst` file by setting `option sysout = on` before the `Solve` statement. It is also normal to specify certain runtime options for the solver within a solver-specific file named `<solver>.opt`.

Download file `ajaxBPDN.gms`. This has the first two solves (and reporting) omitted, but includes the following lines:

```
option sysout      = on;
ajaxBPDN.optfile = 1;
Solve ajaxBPDN using nlp minimizing BPDNobj;
```

Prepare a file called `minos.opt` as follows:

```
LU factor tolerance    2.0
LU update tolerance    2.0
Log frequency          1
Solution                Yes
```

Now type `gams ajaxBPDN nlp=minos` to run the single nonlinear model using MINOS as solver. The aim is to gain experience with controlling GAMS and the solver as well as observing the solver progress. (The MINOS log lines are a bit longer than the default GAMS line-length, but you can edit them a little to give just one line per iteration.)

12. Finally, run GAMS on the BPDN model with SNOPT as solver. The same solver options can be used (they must be in another file called `snopt.opt`). Study the MINOS and SNOPT output files and write some comments about how they compare.

Au revoir

Thank you for being a gentle class. I hope we will meet again every so often. I also hope you encounter lots of matrix problems and optimization problems in the coming years, and that you'll immediately know which software you can put to work on them!

References

- [1] AMPL modeling system. <http://www.ampl.com>.
- [2] E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. on Information Theory*, 52(2):489–509, 2006.
- [3] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Review*, 43(1):129–159, 2001.
- [4] D. L. Donoho. Compressed sensing. *IEEE Trans. on Information Theory*, 52(4):1289–1306, 2006.
- [5] GAMS modeling system. <http://www.gams.com>.
- [6] T. Tao. Preprints in sparse recovery / compressed sensing. <http://www.math.ucla.edu/~tao/preprints/sparse.html>.