Matlab Code – SPSA Algorithm

- The code below implements “Basic” SPSA for iterations k = 1,2,...,n
  - — Global declarations and initialization for program variables `theta`, `alpha`, etc. not shown since that can be handled in numerous ways (e.g., file read, direct inclusion, input during execution)
  - — Δ_k elements are generated by Bernoulli ±1
  - — Program calls an external function “`loss`” to obtain $y(\theta)$ values (noisy loss measurements)
- Simple enhancements possible to increase algorithm stability and/or speed convergence
  - — Check for simple constraint violation (shown at bottom of sample code)
  - — Reject iteration $k \rightarrow k+1$ if $y\left(\hat{\theta}_{k+1}\right)$ is too much greater than $y\left(\hat{\theta}_{k}\right)$ (see Spall (1998)) (requires extra loss measurement per iteration)
  - — Reject iteration $k \rightarrow k+1$ if $\left\|\hat{\theta}_{k+1} - \hat{\theta}_{k}\right\|$ is too large (does not require extra loss measurement)
- Code for the adaptive SPSA algorithm (Spall (2000) sometimes called “2nd – order SPSA”) is available upon request from James Spall.

Matlab Code (cont'd)

Click TXT File button
for editable text.

TXT File

```
for k=0:n-1
   ak=a/(k+1+A)^alpha;
   ck=c/(k+1)^gamma;
   delta=2*round(rand(p,1))-1;
   thetaplus=theta+ck*delta;
   thetaminus=theta-ck*delta;
   yplus=loss(thetaplus);
   yminus=loss(thetaminus);
   ghat=(yplus-yminus)./(2*ck*delta);
   theta=theta-ak*ghat;
end
theta
```

If maximum and minimum values on the values of `theta` can be specified, say `thetamax` and `thetamin`, then the following two lines can be added below the `theta` update line to impose the constraints

```
   theta=min(theta,thetamax);
   theta=max(theta,thetamin);
```

Note: The MATLAB `feval` operation (not used above) is useful in `yplus` and `yminus` evaluations to allow for easy change of loss function.