APPC
TP 9 Stein Unbiased Risk Estimator (SURE)

Thomas Robert

```
load ms_mf

[n, p] = size(Y);
[U, S, V] = svd(Y);
s = diag(S);
```

SURE with SCAD

```
a = 2;
lambdas = 0:0.02:max(s);
SUREs = zeros(size(lambdas));
TrueSUREs = SUREs;
errs = SUREs;

for i = 1:length(lambdas)
    lambda = lambdas(i);
    Mhat = U*SCAD(S, a, lambda)*V';
    errs(i) = sumsqr(Y - Mhat);
    SUREs(i) = errs(i) + (2 * divSURE(S, a, lambda, @SCAD, @SCADderive) - n * p)*sig^2;
    TrueSUREs(i) = sumsqr(M - Mhat);
end

subplot(2,2,1);
hold off;
semilogy(lambdas, SUREs);
hold all;
semilogy(lambdas, TrueSUREs);
ylims = ylim;
semilogy(lambdas, errs);
ylim(ylims);
xlim([lambdas(1) lambdas(end)]);
xlabel('\lambda');
ylabel('Risk / error');
title(['Error for SCAD (\alpha = ' num2str(a) ')']);
```

SURE with AdaLasso

```
a = 1;
lambdas = 0:0.02:max(s);
SUREs = zeros(size(lambdas));
TrueSUREs = SUREs;
errs = SUREs;

for i = 1:length(lambdas)
    lambda = lambdas(i);
    Mhat = U*AdaLasso(S, a, lambda)*V';
    errs(i) = sumsqr(Y - Mhat);
    SUREs(i) = errs(i) + (2 * divSURE(S, a, lambda, @AdaLasso, @AdaLassoDerive) - n * p)*sig^2;
    TrueSUREs(i) = sumsqr(M - Mhat);
end

subplot(2, 2, 2);
hold off;
semilogy(lambdas, SUREs);
hold all;
semilogy(lambdas, TrueSUREs);
```

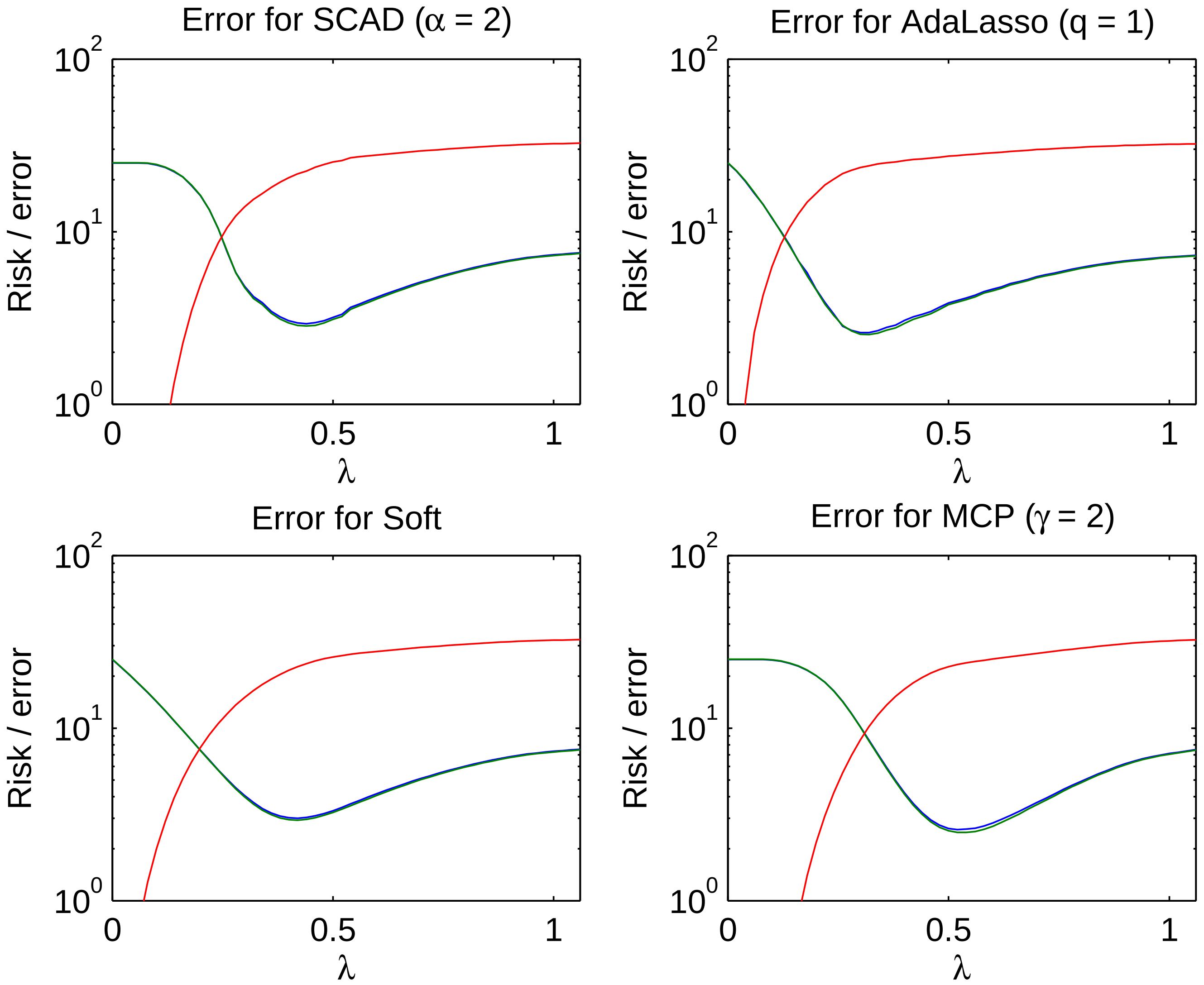
```
ylims = ylim;
semilogy(lambdas, errs);
ylim(ylims);
xlim([lambdas(1) lambdas(end)]);
xlabel('\lambda');
ylabel('Risk / error');
title(['Error for AdaLasso (q = ' num2str(a) ')']);
```

SURE with Soft

```
a = 0; % do not exist
lambdas = 0:0.02:max(s);
SUREs = zeros(size(lambdas));
TrueSUREs = SUREs;
errs = SUREs;

for i = 1:length(lambdas)
    lambda = lambdas(i);
    Mhat = U*Soft(S, a, lambda)*V';
    errs(i) = sumsqr(Y - Mhat);
    SUREs(i) = errs(i) + (2 * divSURE(S, a, lambda, @Soft, @SoftDerive) - n * p)*sig^2;
    TrueSUREs(i) = sumsqr(M - Mhat);
end

subplot(2, 2, 3);
hold off;
semilogy(lambdas, SUREs);
hold all;
semilogy(lambdas, TrueSUREs);
ylims = ylim;
semilogy(lambdas, errs);
ylim(ylims);
xlim([lambdas(1) lambdas(end)]);
xlabel('\lambda');
ylabel('Risk / error');
title('Error for Soft');
```

SURE with MCP

```
a = 2;
lambdas = 0:0.02:max(s);
SUREs = zeros(size(lambdas));
TrueSUREs = SUREs;
errs = SUREs;

for i = 1:length(lambdas)
    lambda = lambdas(i);
    Mhat = U*MCP(S, a, lambda)*V';
    errs(i) = sumsqr(Y - Mhat);
    SUREs(i) = errs(i) + (2 * divSURE(S, a, lambda, @MCP, @MCPDerive) - n * p)*sig^2;
    TrueSUREs(i) = sumsqr(M - Mhat);
end

subplot(2, 2, 4);
hold off;
semilogy(lambdas, SUREs);
hold all;
semilogy(lambdas, TrueSUREs);
ylims = ylim;
semilogy(lambdas, errs);
ylim(ylims);
xlim([lambdas(1) lambdas(end)]);
xlabel('\lambda');
ylabel('Risk / error');
title(['Error for MCP (\gamma = ' num2str(a) ')']);
```


Error for SCAD (α = 2)
Error for AdaLasso (q = 1)
Error for Soft
Error for MCP (γ = 2)
Risk / error
10^2
10^1
10^0
0
0.5
1
λ