```
pr.objective=@tvdenoise;
pr.par=[];
pr.x0=y;
pr.par.data=y;
pr.par.alpha=100;
pr.par.p=2;
pr.par.bc=[0;1];
pr.par.beta=mean(abs(diff(y)))/10000;

pr.method='TrustRegion';
pr.maxiter=1000;
pr.progress=10;
pr.dftol=1E-8;
pr.ngtol=1E-8;
pr.dxtol=1E-8;
pr.delta=0.001;
pr.deltamin=1E-8;
pr.deltamax=100;

out=optimize(pr);
```

```matlab
function [f,g]=tvdenoise(u,par)

% Unpack the parameters
alpha=par.alpha;
beta=par.beta;
p=par.p;
y=par.data;
bc=par.bc;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% compute the data fidelity
duy=u-y;
DataFidelity=(1/2)*sum(duy.^2);

% compute the variation
if isempty(bc)
    ue=[u(1);u;u(end)];
else
    ue=[bc(1);u;bc(2)];
end
du=diff(ue);
Variation=(1/p)*sum(sqrt(du.^2+beta^2).^p);

% compute the objective value
f=DataFidelity+alpha*Variation;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% compute the gradient
if nargout>1
    if p==2
        g=duy-alpha*(du(2:end)-du(1:end-1));
    else
        t2=sqrt(du(1:end-1).^2+beta^2).^(p-2);
        t3=sqrt(du(2:end).^2+beta^2).^(p-2);
        g=duy-alpha*(du(2:end).*t3-du(1:end-1).*t2);
    end
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

return
```