CHAPTER

SPARSE REGRESSION 24

CHAPTER CONTENTS

ℓ_2-constrained LS for linear-in-parameter model,

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{b} \theta_j \phi_j(\boldsymbol{x}),$$

is highly useful in practice. However, when the number of parameters, b, is very large, computing the output value $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ for test sample $\boldsymbol{x}$ is time-consuming. In this chapter, a learning method that tends to produce a *sparse* solution is introduced. A sparse solution means that many of the parameters $\{\theta_j\}_{j=1}^{b}$ take exactly zero, and thus $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ can be computed efficiently.

24.1 ℓ_1-CONSTRAINED LS

ℓ_2-constrained LS uses the ℓ_2-norm as a constraint, while *sparse learning* uses the ℓ_1-norm as a constraint:

$$\min_{\boldsymbol{\theta}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|^2 \ \text{ subject to } \|\boldsymbol{\theta}\|_1 \le R^2, \tag{24.1}$$

where the ℓ_1-norm of vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_b)^\top$ is defined as the absolute sum of all elements:

$$\|\boldsymbol{\theta}\|_1 = \sum_{j=1}^{b} |\theta_j|.$$

An Introduction to Statistical Machine Learning. DOI: 10.1016/B978-0-12-802121-7.00035-2

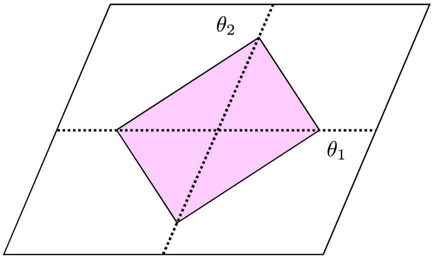


FIGURE 24.1

Parameter space in ℓ_1-constrained LS.

The region that $\|\boldsymbol{\theta}\|_1 \leq R^2$ specifies is illustrated in Fig. 24.1, which has corners on the coordinate axes. This shape is called the ℓ_1-ball, and ℓ_1-constrained LS is also called ℓ_1*-regularization learning*.

Actually, these corners are the key to produce a sparse solution. Let us intuitively explain this idea using Fig. 24.2. For linear-in-parameter model,

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{b} \theta_j \phi_j(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{\phi}(\boldsymbol{x}),$$

the training squared error is a convex quadratic function with respect to $\boldsymbol{\theta}$:

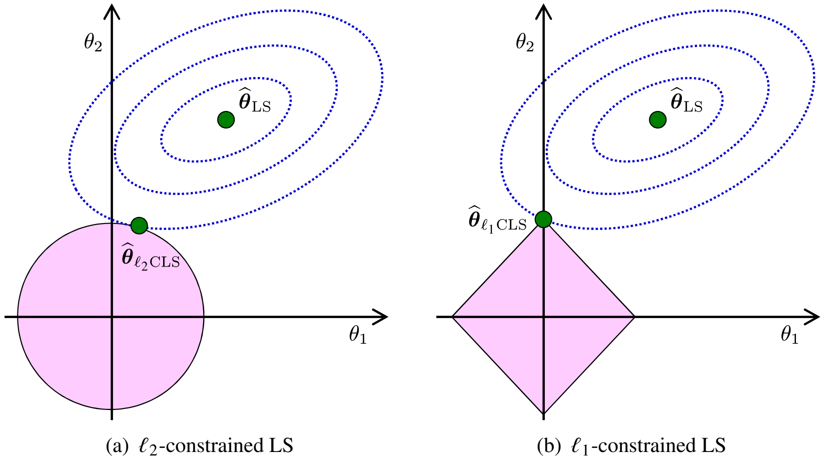
$$J_{\mathrm{LS}}(\boldsymbol{\theta}) = \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|^2.$$

Thus, in the parameter space, J_{LS} has ellipsoidal contours and its minimum is the LS solution $\widehat{\boldsymbol{\theta}}_{\mathrm{LS}}$. As illustrated in Fig. 24.2(a), the solution of ℓ_2-constrained LS is given as the point where the ellipsoidal contour and the ℓ_2-ball touch. Similarly, the solution of ℓ_1-constrained LS is given as the point where the ellipsoidal contour and the ℓ_1-ball touch. Since the ℓ_1-ball has corners on the coordinate axes, the solution tends to be at one of the corners, which is a sparse solution (see Fig. 24.2(b)).

In statistics, ℓ_1-constrained LS is referred to as *lasso regression* [110].

24.2 SOLVING ℓ_1-CONSTRAINED LS

Since the ℓ_1-norm is not differentiable at the origin, solving ℓ_1-constrained LS problem (24.1) is not as straightforward as ℓ_2-constrained LS. In this section, a general optimization technique called the *alternating direction method of multipliers* (ADMM) [20] is applied to ℓ_1-constrained LS and gives a simple yet practical algorithm. The algorithm of ADMM is summarized in Fig. 24.3.


θ_2
$\widehat{\boldsymbol{\theta}}_{\mathrm{LS}}$
$\widehat{\boldsymbol{\theta}}_{\ell_2\mathrm{CLS}}$
θ_1
θ_2
$\widehat{\boldsymbol{\theta}}_{\mathrm{LS}}$
$\widehat{\boldsymbol{\theta}}_{\ell_1\mathrm{CLS}}$
θ_1

(a) ℓ_2-constrained LS

(b) ℓ_1-constrained LS

FIGURE 24.2

The solution of ℓ_1-constrained LS tends to be on one of the coordinate axes, which is a sparse solution.

Let us express the optimization problem of ℓ_1-constrained LS, Eq. (24.1), as

$$\min_{\boldsymbol{\theta},\boldsymbol{z}} \left[\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}\|^2 + \lambda\|\boldsymbol{z}\|_1\right] \text{ subject to } \boldsymbol{\theta} = \boldsymbol{z},$$

where $\lambda \geq 0$. The *augmented Lagrange function* for this optimization problem is given by

$$L(\boldsymbol{\theta},\boldsymbol{z},\boldsymbol{u}) = \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}\|^2 + \lambda\|\boldsymbol{z}\|_1 + \boldsymbol{u}^\top(\boldsymbol{\theta}-\boldsymbol{z}) + \frac{1}{2}\|\boldsymbol{\theta}-\boldsymbol{z}\|^2,$$

where $\boldsymbol{u}$ is the *Lagrange multipliers*. Since

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = -\boldsymbol{\Phi}^\top(\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}) + \boldsymbol{u} + \boldsymbol{\theta} - \boldsymbol{z},$$

setting this to zero yields the following update equation for $\boldsymbol{\theta}$:

$$\begin{aligned}\boldsymbol{\theta}^{(k+1)} &= \operatorname*{argmin}_{\boldsymbol{\theta}} L(\boldsymbol{\theta},\boldsymbol{z}^{(k)},\boldsymbol{u}^{(k)})\\ &= (\boldsymbol{\Phi}^\top\boldsymbol{\Phi}+\boldsymbol{I})^{-1}(\boldsymbol{\Phi}^\top\boldsymbol{y}+\boldsymbol{z}^{(k)}-\boldsymbol{u}^{(k)}).\end{aligned}$$

From

$$\begin{aligned}&\min_{z}\left[\lambda|z| + u(\theta - z) + \frac{1}{2}(\theta-z)^2\right]\\ &\quad = \max(0,\theta+u-\lambda) + \min(0,\theta+u+\lambda),\end{aligned}$$

The ADMM algorithm for solving the constrained optimization problem,

$$\min_{\boldsymbol{\theta},\boldsymbol{z}} [f(\boldsymbol{\theta}) + g(\boldsymbol{z})] \quad \text{subject to } \boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{B}\boldsymbol{z} = \boldsymbol{c},$$

is given as follows:

$$\begin{aligned}
\boldsymbol{\theta}^{(k+1)} &= \operatorname*{argmin}_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, \boldsymbol{z}^{(k)}, \boldsymbol{u}^{(k)}),\\
\boldsymbol{z}^{(k+1)} &= \operatorname*{argmin}_{\boldsymbol{z}} L(\boldsymbol{\theta}^{(k+1)}, \boldsymbol{z}, \boldsymbol{u}^{(k)}),\\
\boldsymbol{u}^{(k+1)} &= \boldsymbol{u}^{(k)} + \boldsymbol{A}\boldsymbol{\theta}^{(k+1)} + \boldsymbol{B}\boldsymbol{z}^{(k+1)} - \boldsymbol{c},
\end{aligned}$$

where $\boldsymbol{u}$ is the *Lagrange multipliers* and L is the *augmented Lagrange function* defined as

$$\begin{aligned}
L(\boldsymbol{\theta}, \boldsymbol{z}, \boldsymbol{u}) = f(\boldsymbol{\theta}) + g(\boldsymbol{z}) + \boldsymbol{u}^\top(\boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c})\\
+ \frac{1}{2}\|\boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}\|^2.
\end{aligned}$$

An advantage of ADMM is that no tuning parameter such as the step size in the gradient algorithm (see Fig. 22.6) is involved.

FIGURE 24.3

Alternating direction method of multipliers.

the update equation for $\boldsymbol{z}$ is given as

$$\begin{aligned}
\boldsymbol{z}^{(k+1)} &= \operatorname*{argmin}_{\boldsymbol{z}} L(\boldsymbol{\theta}^{(k+1)}, \boldsymbol{z}, \boldsymbol{u}^{(k)})\\
&= \max(\mathbf{0}, \boldsymbol{\theta}^{(k+1)} + \boldsymbol{u}^{(k)} - \lambda\mathbf{1}) + \min(\mathbf{0}, \boldsymbol{\theta}^{(k+1)}\boldsymbol{u}^{(k)} + \lambda\mathbf{1}),
\end{aligned}$$

where $\mathbf{1}$ is the vector with all ones. Finally, the update equation for $\boldsymbol{u}$ is given as follows (see Fig. 24.3):

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + \boldsymbol{\theta}^{(k+1)} - \boldsymbol{z}^{(k+1)}.$$

A MATLAB code of ℓ_1-constrained LS by ADMM for the Gaussian kernel model,

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{n} \theta_j \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}_j\|^2}{2h^2}\right),$$

is provided in Fig. 24.4, and its behavior is illustrated in Fig. 24.5. The obtained solution is actually very similar to the one obtained by ℓ_2-constrained LS (see

```
n=50; x=linspace(-3,3,n)'; pix=pi*x;
y=sin(pix)./(pix)+0.1*x+0.2*randn(n,1);

hh=2*0.3^2; l=0.1; x2=x.^2;
k=exp(-(repmat(x2,1,n)+repmat(x2',n,1)-2*x*x')/hh);
ky=k*y; A=inv(k^2+eye(n)); t0=zeros(n,1); z=t0; u=t0;
for o=1:1000
 t=A*(ky+z-u); z=max(0,t+u-l)+min(0,t+u+l); u=u+t-z;
 if norm(t-t0)<0.0001, break, end
 t0=t;
end

N=1000; X=linspace(-3,3,N)';
K=exp(-(repmat(X.^2,1,n)+repmat(x2',N,1)-2*X*x')/hh); F=K*t;
figure(1); clf; hold on; axis([-2.8 2.8 -1 1.5]);
plot(X,F,'g-'); plot(x,y,'bo'); sum(abs(t)<0.001)
```

FIGURE 24.4

MATLAB code of ℓ_1-constrained LS by ADMM for Gaussian kernel model.

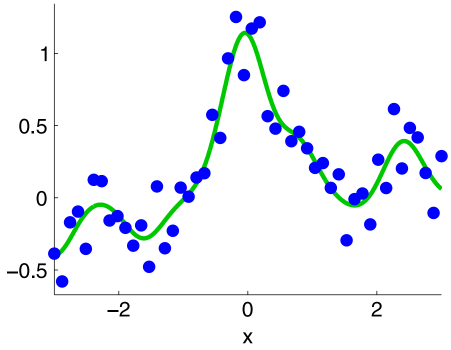


FIGURE 24.5

Example of ℓ_1-constrained LS for Gaussian kernel model. 38 out of 50 parameters are zero.

Fig. 23.7). However, 38 out of 50 parameters are zero in ℓ_1-constrained LS, while all 50 parameters are nonzero in ℓ_2-constrained LS.

When the *stochastic gradient* algorithm (see Fig. 22.6) is used to obtain the LS solution for linear-in-parameter model,

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{b} \theta_j \phi_j(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{\phi}(\boldsymbol{x}),$$

parameter $\boldsymbol{\theta}$ is updated as

$$\boldsymbol{\theta} \longleftarrow \boldsymbol{\theta} + \varepsilon \boldsymbol{\phi}(\boldsymbol{x})\big(y - f_{\boldsymbol{\theta}}(\boldsymbol{x})\big),$$

where $\varepsilon > 0$ is the step size and $(\boldsymbol{x}, y)$ is a randomly chosen training sample. To obtain a sparse solution by the stochastic gradient algorithm, parameter $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_b)^\top$ may be thresholded once in every several gradient updates as follows [66]:

$$\forall j = 1, \ldots, b, \quad \theta_j \longleftarrow \begin{cases} \max(0, \theta_j - \lambda\varepsilon) & (\theta_j > 0), \\ \min(0, \theta_j + \lambda\varepsilon) & (\theta_j \leq 0). \end{cases}$$

24.3 FEATURE SELECTION BY SPARSE LEARNING

Let us apply sparse learning to the linear-in-input model for $\boldsymbol{x} = (x^{(1)}, \ldots, x^{(d)})^\top$:

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{d} \theta_j x^{(j)} = \boldsymbol{\theta}^\top \boldsymbol{x}.$$

If $\theta_j = 0$, then the jth input variable $x^{(j)}$ does not appear in the final prediction model. Thus, *feature selection* can be performed by sparse learning.

For example, suppose that the income of a person is modeled by

$$\theta_1 \times \text{Education} + \theta_2 \times \text{Age} + \theta_3 \times \text{Ability} + \theta_4 \times \text{Parents' income},$$

and sparse learning gives $\theta_4 = 0$. This means that parents' income is not related to the income of the child.

If such feature selection is naively performed, all 2^d combinations of d features $x^{(1)}, \ldots, x^{(d)}$ need to be investigated, which is not tractable when d is large. In practice, greedy strategies such as *forward selection* of adding a feature one by one and *backward elimination* of deleting a feature one by one are often used. However, since such greedy approaches do not consider the dependency between features, they do not necessarily give good feature combinations. If sparse learning based on the ℓ_1-norm is used, dependency between features can be taken into account to some extent. However, feature selection by sparse learning is possible only for the *linear-in-input model*. If *linear-in-parameter models* are used, just a subset of basis functions is selected, which is different from feature selection because every basis function depends on all features in general.

24.4 VARIOUS EXTENSIONS

In this section, various extensions to ℓ_1-constrained LS are introduced.

24.4.1 GENERALIZED ℓ_1-CONSTRAINED LS

For some matrix $\boldsymbol{F}$, *generalized ℓ_1-constrained LS* is defined as

$$\min_{\boldsymbol{\theta}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|^2 \ \text{ subject to } \|\boldsymbol{F}\boldsymbol{\theta}\|_1 \leq R^2,$$

and its ADMM form is given by

$$\min_{\boldsymbol{\theta},\boldsymbol{z}} \left[\frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|^2 + \lambda\|\boldsymbol{z}\|_1\right] \ \text{ subject to } \boldsymbol{F}\boldsymbol{\theta} = \boldsymbol{z}.$$

Then the ADMM update formulas are given by

$$\begin{aligned}
\boldsymbol{\theta}^{(k+1)} &= (\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + \boldsymbol{F}^\top\boldsymbol{F})^{-1}(\boldsymbol{\Phi}^\top\boldsymbol{y} + \boldsymbol{F}^\top\boldsymbol{z}^{(k)} - \boldsymbol{F}^\top\boldsymbol{u}^{(k)}),\\
\boldsymbol{z}^{(k+1)} &= \max(\boldsymbol{0}, \boldsymbol{F}\boldsymbol{\theta}^{(k+1)} + \boldsymbol{u}^{(k)} - \lambda\boldsymbol{1}) + \min(\boldsymbol{0}, \boldsymbol{F}\boldsymbol{\theta}^{(k+1)}\boldsymbol{u}^{(k)} + \lambda\boldsymbol{1}),\\
\boldsymbol{u}^{(k+1)} &= \boldsymbol{u}^{(k)} + \boldsymbol{F}\boldsymbol{\theta}^{(k+1)} - \boldsymbol{z}^{(k+1)}.
\end{aligned}$$

A notable example of generalized ℓ_1-constrained LS in statistics is *fused lasso* [111]:

$$\min_{\boldsymbol{\theta}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|^2 \ \text{ subject to } \sum_{j=1}^{b-1} |\theta_{j+1} - \theta_j| \leq R^2,$$

which corresponds to

$$F_{j,j'} = \begin{cases} 1 & (j' = j+1),\\ -1 & (j' = j),\\ 0 & (\text{otherwise}). \end{cases}$$

When $\boldsymbol{\Phi}$ is the identity matrix, this problem is called the *total variation denoising* in the signal processing community.

24.4.2 ℓ_p-CONSTRAINED LS

Let us generalize ℓ_2-constrained LS and ℓ_1-constrained LS to ℓ_p-constrained LS for $p \geq 0$:

$$\|\boldsymbol{\theta}\|_p \leq R^2.$$

Here, the *ℓ_p-norm* of vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_b)^\top$ is defined as

$$\|\boldsymbol{\theta}\|_p = \begin{cases} \sum_{j=1}^{b} \delta(\theta_j \neq 0) & (p = 0),\\ \left(\sum_{j=1}^{b} |\theta_j|^p\right)^{\frac{1}{p}} & (0 < p < \infty),\\ \max\left\{|\theta_1|, \ldots, |\theta_b|\right\} & (p = \infty), \end{cases}$$

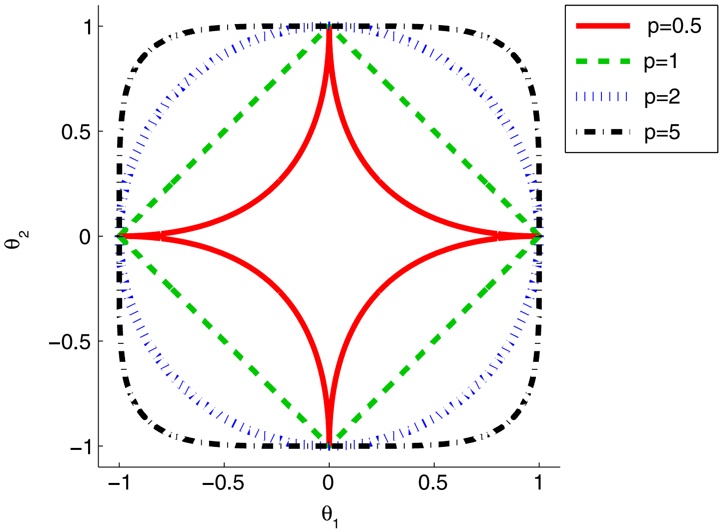


FIGURE 24.6

Unit ℓ_p-balls.

where

$$\delta(\theta_j \neq 0) = \begin{cases} 1 & (\theta_j \neq 0), \\ 0 & (\theta_j = 0). \end{cases}$$

Thus, the ℓ_0-norm gives the number of nonzero elements, and the ℓ_∞-norm gives the maximum element.

Fig. 24.6 shows the unit ℓ_p-balls (i.e. $\|\boldsymbol{\theta}\|_p = 1$) for $p = 0.5$, 1, 2, and 5. The ℓ_p-ball has corners on the coordinate axes when $p \leq 1$, and the ℓ_p-ball has a convex shape when $p \geq 1$. As illustrated in Fig. 24.2, having corners on the coordinate axes is the key to produce a sparse solution. On the other hand, having a convex shape is essential to obtain the global optimal solution. Thus, $p = 1$ is the best choice that only allows sparsity induction under the convex formulation (see Fig. 24.7).

24.4.3 $\ell_1 + \ell_2$-CONSTRAINED LS

ℓ_1-constrained LS is a useful method in practice, but it has several drawbacks.

When the number of parameters, b, is larger than the number of training samples, n, the number of nonzero values in the solution of ℓ_1-constrained LS is at most n.

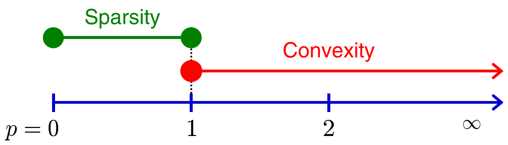


FIGURE 24.7

Properties of ℓ_p-constraint.

This is not a problem when a kernel model is used, because $b = n$:

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{n} \theta_j K(\boldsymbol{x}, \boldsymbol{x}_j).$$

However, this could be a critical limitation when feature selection is performed using the linear-in-input model (see Section 24.3),

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{d} \theta_j x^{(j)} = \boldsymbol{\theta}^\top \boldsymbol{x},$$

because only n features can be selected at most.

When several basis functions are similar to each other and having group structure, ℓ_1-constrained LS tends to choose only one of them and ignore the rest. This means that, when feature selection is performed using the linear-in-input model, only one feature is chosen from a group of correlated features. When kernel models are used, the kernel basis functions tend to have group structure, if training input samples $\{\boldsymbol{x}_i\}_{i=1}^n$ have cluster structure.

$\ell_1 + \ell_2$*-constrained LS* can overcome the above drawbacks of ℓ_1-constrained LS, which uses the sum of the ℓ_1-norm and the ℓ_2-norm as a constraint:

$$(1 - \tau)\|\boldsymbol{\theta}\|_1 + \tau\|\boldsymbol{\theta}\|^2 \leq R^2,$$

where $0 \leq \tau \leq 1$ controls the balance between the ℓ_1-norm and ℓ_2-norm constraints. The $(\ell_1 + \ell_2)$-constraint is reduced to the ℓ_1-constraint if $\tau = 0$, and the $(\ell_1 + \ell_2)$-constraint is reduced to the ℓ_2-constraint if $\tau = 1$. When $0 \leq \tau < 1$, the region $(1 - \tau)\|\boldsymbol{\theta}\|_1 + \tau\|\boldsymbol{\theta}\|^2 \leq R^2$ has corners on the coordinate axes.

Fig. 24.8 illustrates the unit $(\ell_1 + \ell_2)$-ball for balance parameter $\tau = 1/2$, which is similar to the unit $\ell_{1.4}$-ball. However, while the $\ell_{1.4}$-ball has no corner as the ℓ_2-ball, the $(\ell_1 + \ell_2)$-ball has corners. Thus, the $(\ell_1 + \ell_2)$-constraint tends to produce a sparse solution.

Let us express the optimization problem of $(\ell_1 + \ell_2)$-constrained LS as

$$\min_{\boldsymbol{\theta}, \boldsymbol{z}} \left[\frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\theta}\|^2 + \eta\|\boldsymbol{\theta}\|^2 + \lambda\|\boldsymbol{z}\|_1 \right] \quad \text{subject to } \boldsymbol{\theta} = \boldsymbol{z},$$

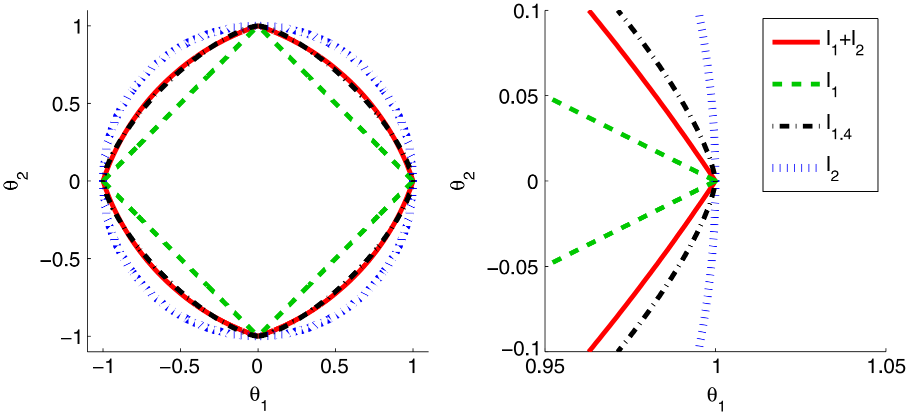


FIGURE 24.8

Unit $(\ell_1 + \ell_2)$-norm ball for balance parameter $\tau = 1/2$, which is similar to the unit $\ell_{1.4}$-ball. However, while the $\ell_{1.4}$-ball has no corner, the $(\ell_1 + \ell_2)$-ball has corners.

where $\lambda, \eta \geq 0$. Then the solution of $(\ell_1 + \ell_2)$-constrained LS can be obtained almost in the same way as ℓ_1-constrained LS by ADMM (see Section 24.2), but only the update equation for $\boldsymbol{\theta}$ is changed as

$$\boldsymbol{\theta}^{(k+1)} = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + (\eta + 1)\boldsymbol{I})^{-1}(\boldsymbol{\Phi}^\top \boldsymbol{y} + \boldsymbol{z}^{(k)} - \boldsymbol{u}^{(k)}).$$

Even when the number of parameters is larger than the number of training samples, i.e. $b > n$, $(\ell_1 + \ell_2)$-constrained LS can produce a solution with more than n nonzero elements. Furthermore, $(\ell_1 + \ell_2)$-constrained LS tends to choose features in a groupwise manner, i.e. all features in the same group tend to be discarded simultaneously. However, in addition to the regularization parameter λ, the balance parameter τ needs to be tuned, which is cumbersome in practice.

In statistics, $(\ell_1 + \ell_2)$-constrained LS is referred to as *elastic-net regression* [124].

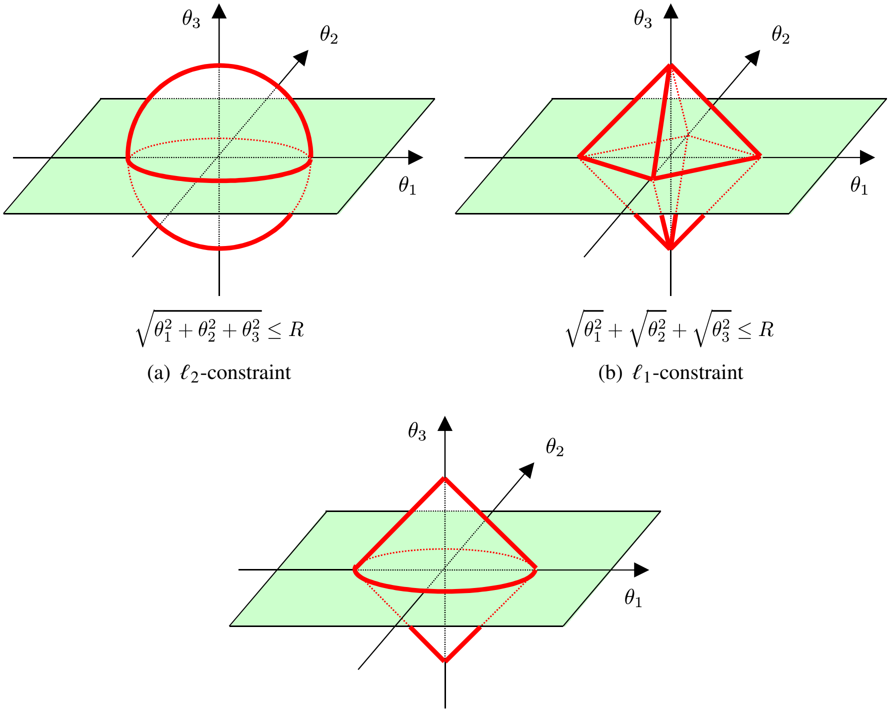
24.4.4 $\ell_{1,2}$-CONSTRAINED LS

Suppose that the parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_b)^\top$ has group structure as

$$\boldsymbol{\theta} = (\boldsymbol{\theta}^{(1)\top}, \ldots, \boldsymbol{\theta}^{(t)\top})^\top,$$

where $\boldsymbol{\theta}^{(j)} \in \mathbb{R}^{b_j}$ and $\sum_{j=1}^{t} b_j = b$. Then, LS with $\ell_{1,2}$-constraint,

$$\sum_{j=1}^{t} \|\boldsymbol{\theta}^{(j)}\| \leq R,$$



$\sqrt{\theta_1^2+\theta_2^2+\theta_3^2} \leq R$

(a) ℓ_2-constraint

$\sqrt{\theta_1^2}+\sqrt{\theta_2^2}+\sqrt{\theta_3^2} \leq R$

(b) ℓ_1-constraint

$\sqrt{\theta_1^2+\theta_2^2}+\sqrt{\theta_3^2} \leq R$

(c) $\ell_{1,2}$-constraint

FIGURE 24.9

Constraints in three-dimensional parameter space.

tends to give a groupwise sparse solution, i.e. some of parameter subvector $\boldsymbol{\theta}^{(j)}$ becomes zero. Such a learning method is called $\ell_{1,2}$-*constrained LS*.

The ℓ_2-constraint, ℓ_1-constraint, and $\ell_{1,2}$-constraint in three-dimensional parameter space are illustrated in Fig. 24.9. The $\ell_{1,2}$-region consists of the ℓ_2-part and the ℓ_1-part, and if the solution is on the top/bottom peak of the $\ell_{1,2}$-region, it will be sparse as $(\theta_1, \theta_2, \theta_3) = (0, 0, \pm R)$. On the other hand, if the solution is on the circle in the (θ_1, θ_2)-plane, the solution will be sparse as $(\theta_1, \theta_2, \theta_3) = (a, b, 0)$ for $a^2 + b^2 = R^2$.

In statistics, $\ell_{1,2}$-constrained LS is referred to as *group-lasso regression* [122]. The $\ell_{1,2}$-constraint plays an important role in advanced machine learning topics such as *multitask feature selection* (Section 34.3), *structural change detection* (Section 39.2.2), and *multiple kernel learning* [11].

The *trace norm* of matrix $\mathbf{\Theta} \in \mathbb{R}^{d_1 \times d_2}$, denoted by $\|\mathbf{\Theta}\|_{\mathrm{tr}}$, is defined as

$$\|\mathbf{\Theta}\|_{\mathrm{tr}} = \sum_{k=1}^{\min(d_1,d_2)} \sigma_k,$$

where σ_k is a *singular value* of $\mathbf{\Theta}$ (see Fig. 22.2). The trace norm is also referred to as the *nuclear norm*. Given that singular values are non-negative, the trace norm $\|\mathbf{\Theta}\|_{\mathrm{tr}}$ can be regarded as the ℓ_1-norm on singular values. Thus, using the trace norm $\|\mathbf{\Theta}\|_{\mathrm{tr}}$ as a regularizer tends to produce a *sparse* solution on singular values (see Chapter 24), implying that $\mathbf{\Theta}$ becomes a low-rank matrix. Note that if $\mathbf{\Theta}$ is squared and diagonal, $\|\mathbf{\Theta}\|_{\mathrm{tr}}$ is reduced to the ℓ_1-norm of the diagonal entries. Thus, the trace norm can be regarded as an extension of the ℓ_1-norm from vectors to matrices.

FIGURE 24.10

Trace norm of a matrix.

24.4.5 TRACE NORM CONSTRAINED LS

So far, data samples $\{\boldsymbol{x}_i\}_{i=1}^n$ are assumed to be d-dimensional vectors. However, in some applications such as image analysis, a sample $\boldsymbol{x}_i$ may be a matrix (corresponding to a two-dimensional image). Such matrix samples can be naively handled if they are vectorized (see Fig. 6.5), but then the two-dimensional structure of images is completely lost. Here, a regularization technique for matrix samples is introduced [112].

Suppose that matrix-input scalar-output paired samples $\{(\boldsymbol{X}_i, y_i)\}_{i=1}^n$ are given as training data, where $\boldsymbol{X}_i \in \mathbb{R}^{d_1 \times d_2}$ and $y_i \in \mathbb{R}$. Let us employ the *linear-in-input* model for parameter matrix $\mathbf{\Theta} \in \mathbb{R}^{d_1 \times d_2}$,

$$f_{\mathbf{\Theta}}(\boldsymbol{X}) = \mathrm{tr}\left(\mathbf{\Theta}^\top \boldsymbol{X}\right),$$

which is equivalent to the vectorized linear-in-input model $\mathrm{vec}(\mathbf{\Theta})^\top \mathrm{vec}(\boldsymbol{X})$. To utilize the two-dimensional structure, the *trace norm* $\|\mathbf{\Theta}\|_{\mathrm{tr}}$ is employed as a constraint (see Fig. 24.10):

$$\min_{\mathbf{\Theta} \in \mathbb{R}^{d_1 \times d_2}} \frac{1}{2} \sum_{i=1}^n \left(y_i - f_{\mathbf{\Theta}}(\boldsymbol{X}_i)\right)^2 \text{ subject to } \|\mathbf{\Theta}\|_{\mathrm{tr}} \leq R.$$

Note that this optimization problem is convex, and thus the global optimal solution can be easily obtained, e.g, by the *proximal gradient method* (see Fig. 34.8). Thanks to the trace norm constraint, the solution of $\mathbf{\Theta}$ tends to be a low-rank matrix.