Note on Linear Regression Gradients
Christos Giannakopoulos

1 Deriving Linear Regression via Least Squares

We are given data points (x_i, y_i) and assume a linear model:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

In matrix form:

$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon} \quad \text{where} \quad X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}$$

We want to minimize the residual sum of squares (RSS):

$$\chi^2(\boldsymbol{\beta}) = \|\mathbf{y} - X\boldsymbol{\beta}\|^2 = (\mathbf{y} - X\boldsymbol{\beta})^\top (\mathbf{y} - X\boldsymbol{\beta})$$

Expanding:

$$\chi^2(\boldsymbol{\beta}) = \mathbf{y}^\top \mathbf{y} - 2\boldsymbol{\beta}^\top X^\top \mathbf{y} + \boldsymbol{\beta}^\top X^\top X \boldsymbol{\beta}$$

Taking the derivative:

$$\frac{d}{d\boldsymbol{\beta}} \chi^2(\boldsymbol{\beta}) = -2X^\top \mathbf{y} + 2X^\top X\boldsymbol{\beta}$$

Setting the gradient to zero:

$$\boxed{X^\top X\boldsymbol{\beta} = X^\top \mathbf{y} \quad \Rightarrow \quad \hat{\boldsymbol{\beta}} = (X^\top X)^{-1} X^\top \mathbf{y}}$$

This gives the best-fit parameters that minimize squared error.

Derivation

Consider the quadratic form:

$$f(\boldsymbol{\beta}) = \boldsymbol{\beta}^\top A \boldsymbol{\beta}$$

We want to compute the gradient:

$$\nabla_{\boldsymbol{\beta}} f = \frac{d}{d\boldsymbol{\beta}} \left(\boldsymbol{\beta}^\top A \boldsymbol{\beta} \right)$$

First, expand the quadratic form:

$$f(\boldsymbol{\beta}) = \sum_{i=1}^{n} \sum_{j=1}^{n} \beta_i a_{ij} \beta_j$$

Now compute the partial derivative with respect to β_k:

$$\frac{\partial f}{\partial \beta_k} = \sum_{j=1}^{n} a_{kj} \beta_j + \sum_{i=1}^{n} a_{ik} \beta_i$$

When A is symmetric, i.e., $a_{ij} = a_{ji}$, this simplifies to:

$$\frac{\partial f}{\partial \beta_k} = 2\sum_{j=1}^{n} a_{kj}\beta_j$$

Thus, the gradient is:

$$\nabla_{\boldsymbol{\beta}} f = 2A\boldsymbol{\beta}$$

2 Geometric Interpretation

Case 1: $A = I$

$$f(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|^2 \quad \Rightarrow \quad \nabla f = 2\boldsymbol{\beta}$$

The contours are circles (or spheres) centered at the origin. The gradient points radially outward.

Case 2: A Diagonal

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \quad \Rightarrow \quad f(\boldsymbol{\beta}) = 2\beta_1^2 + \beta_2^2$$

Contours are ellipses aligned with the coordinate axes.

Case 3: General Symmetric A

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

Contours are rotated ellipses. The principal axes are given by the eigenvectors of A.

3 Python Code for Visualization

```python
import numpy as np
import matplotlib.pyplot as plt

# Grid setup
beta1 = np.linspace(-2, 2, 100)
beta2 = np.linspace(-2, 2, 100)
B1, B2 = np.meshgrid(beta1, beta2)

# Define A matrices
A_identity = np.array([[1, 0], [0, 1]])
A_diagonal = np.array([[2, 0], [0, 1]])
A_general = np.array([[2, 1], [1, 2]])

def quadratic_form(A, B1, B2):
    return A[0,0]*B1**2 + (A[0,1]+A[1,0])*B1*B2 + A[1,1]*B2**2

# Compute forms
F_identity = quadratic_form(A_identity, B1, B2)
F_diagonal = quadratic_form(A_diagonal, B1, B2)
F_general = quadratic_form(A_general, B1, B2)

# Plotting
fig, axes = plt.subplots(1, 3, figsize=(18, 5))
titles = ["A = Identity", "A = Diagonal", "A = General Symmetric"]
Fs = [F_identity, F_diagonal, F_general]
```

```
As = [A_identity, A_diagonal, A_general]

for ax, F, A, title in zip(axes, Fs, As, titles):
    contour = ax.contour(B1, B2, F, levels=20, cmap=’viridis’)
    ax.clabel(contour, inline=True, fontsize=8)
    grad1 = 2*(A[0,0]*B1 + A[0,1]*B2)
    grad2 = 2*(A[1,0]*B1 + A[1,1]*B2)
    ax.quiver(B1, B2, grad1, grad2, color=’red’, alpha=0.6, scale=40)
    ax.set_title(title)
    ax.set_xlabel(r’$\beta_1$’)
    ax.set_ylabel(r’$\beta_2$’)
    ax.set_aspect(’equal’)
    ax.grid(True)

plt.tight_layout()
plt.show()
```

4 Simple Case Plots

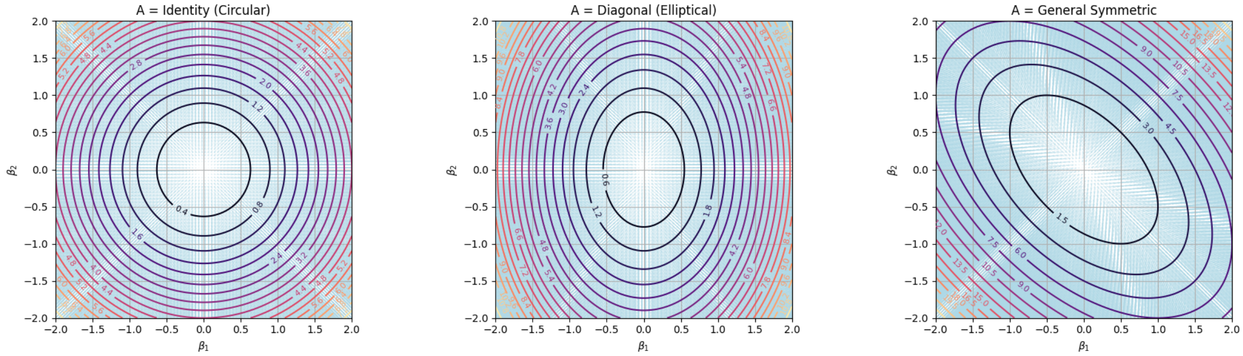

Figure 1: Contours and gradient fields for different matrices A for the 3 different cases. The main conclusion is that when performing Least Squares minimization using the χ^2 for estimator, the parameter space is approximated by a quadratic form for which the gradient gives us the direction of steepest ascent which tells us where the ”bottom” of that quadratic form is.