Nonlinear Regression I

Univariate Scalar Models

Uwe Naumann

Informatik 12:
Software and Tools for Computational Engineering

RWTH Aachen University

Contents

Outline

Objective

- Introduction to nonlinear regression methods for univariate scalar models.

Learning Outcomes

- You will understand
  - normal equation
  - Givens rotation
  - Householder reflection

  in the context of linearization of nonlinear regression problems.
- You will be able to
  - implement nonlinear regression methods
  - run computational experiments.

Outline

The calibration of nonlinear (in $p \in \boldsymbol{R}$) models to given data $(\mathbf{x}, \mathbf{y}) \in \boldsymbol{R}^m \times \boldsymbol{R}^m$ can be posed as a nonlinear minimization problem with objective

$$E(p) = \|F(p, \mathbf{x}, \mathbf{y})\|_2^2 = \sum_{i=0}^{m-1} F_i(p, \mathbf{x}, \mathbf{y})^2 = \sum_{i=0}^{m-1} (f(p, x_i) - y_i)^2 \; .$$

Potential solution methods include steepest gradient descent as well as bisection or Newton's method applied to the first-order optimality criterion $\frac{dE(p)}{dp} = 0$ while satisfying the second-order optimality criterion $\frac{d^2E(p)}{dp^2} > 0$.

These general-purpose algorithms approximate the solution iteratively up to a given accuracy. Such iteration can be avoided in the linear (in p) case yielding a lower computational complexity of linear regression methods.

Linearization allows for application of these ideas to the nonlinear case.

Formulation of the first-order optimality condition

$$E'(p) \equiv \frac{dE(p)}{dp} = \frac{d\|F(p,\mathbf{x},\mathbf{y})\|_2^2}{dp} = 0$$

in terms of a linearization (linearity in Δp) of the residual $F(p,\mathbf{x},\mathbf{y})$ as

$$\frac{d\|F(p,\mathbf{x},\mathbf{y}) + \frac{dF(p,\mathbf{x},\mathbf{y})}{dp} \cdot \Delta p\|_2^2}{d\Delta p} = \frac{d\|F(p,\mathbf{x},\mathbf{y}) + F'(p,\mathbf{x},\mathbf{y}) \cdot \Delta p\|_2^2}{d\Delta p} = 0$$

yields a [damped] iterative optimization scheme for p as

$$p := p + [\alpha \cdot]\Delta p$$

for $0 < \alpha \leq 1$. The Jacobian $F'(p,\mathbf{x},\mathbf{y}) \in R^m$ is required in addition to $F(p,\mathbf{x},\mathbf{y}) \in R^m$. It can be computed symbolically as well as by finite difference approximation or by algorithmic differentiation.

The linear regression problem

$$F'(p,\mathbf{x},\mathbf{y}) \cdot \Delta p \approx -F(p,\mathbf{x},\mathbf{y}) \;, \quad F, F' \in \boldsymbol{R}^m, \;\Delta p \in \boldsymbol{R}$$

can be solved with

- the normal equations method
- Givens rotation
- Householder reflection.

Convergence of the fixed-point iteration $p = G(p) = p + \Delta p$ requires

$$\|G'(p)\| < 1$$

at the solution p^* implying existence of a neighborhood of p^* containing values of p for which the fixed-point iteration converges to this solution.

Outline

The linear regression problem

$$F'(p,\mathbf{x},\mathbf{y}) \cdot \Delta p \approx -F(p,\mathbf{x},\mathbf{y})$$

yields the normal equation

$$F'(p,\mathbf{x},\mathbf{y})^T \cdot F'(p,\mathbf{x},\mathbf{y}) \cdot \Delta p = -F'(p,\mathbf{x},\mathbf{y})^T \cdot F(p,\mathbf{x},\mathbf{y})$$

and hence the solution

$$\Delta p := -\frac{F'(p,\mathbf{x},\mathbf{y})^T \cdot F(p,\mathbf{x},\mathbf{y})}{F'(p,\mathbf{x},\mathbf{y})^T \cdot F'(p,\mathbf{x},\mathbf{y})} .$$

With $\mathbf{a} \equiv F'(p, \mathbf{x}, \mathbf{y})$ and $\mathbf{b} = -F(p, \mathbf{x}, \mathbf{y})$ we get

$$0 = \frac{d\|\mathbf{a} \cdot \Delta p - \mathbf{b}\|_2^2}{d\Delta p} = \frac{d\left[\sum_{i=0}^{m-1}(a_i \cdot \Delta p - b_i)^2\right]}{d\Delta p}$$

$$= 2 \cdot \sum_{i=0}^{m-1} a_i \cdot (a_i \cdot \Delta p - b_i) = \sum_{i=0}^{m-1} a_i^2 \cdot \Delta p - a_i \cdot b_i$$

$$= \Delta p \cdot \sum_{i=0}^{m-1} a_i^2 - \sum_{i=0}^{m-1} a_i \cdot b_i = \Delta p \cdot \mathbf{a}^T \cdot \mathbf{a} - \mathbf{a}^T \cdot \mathbf{b}$$

implying

$$\Delta p = \frac{\mathbf{a}^T \cdot \mathbf{b}}{\mathbf{a}^T \cdot \mathbf{a}} = -\frac{F'(p, \mathbf{x}, \mathbf{y})^T \cdot F(p, \mathbf{x}, \mathbf{y})}{F'(p, \mathbf{x}, \mathbf{y})^T \cdot F'(p, \mathbf{x}, \mathbf{y})}.$$

```
using namespace Eigen;

template<typename T, int M>
T NormalEquation(const Matrix<T,M,1> &drdp, const Matrix<T,M,1> &r) {
  return drdp.dot(r)/drdp.dot(drdp);
}

template<typename T, int M>
T Regression(const Matrix<T,M,1> &x, const Matrix<T,M,1> &y) {
  T p=1;
  while (fabs(dEdp(p,x,y))>1e-7) {
    Matrix<T,M,1> r=F(p,x,y), drdp=dFdp(p,x);
    p-=NormalEquation(drdp,r);
  }
  return p;
}
```

Outline

Solution of the linear regression problem

$$F'(p, \mathbf{x}, \mathbf{y}) \cdot \Delta p \approx -F(p, \mathbf{x}, \mathbf{y})$$

by Givens rotation transforms the vector

$$F'(p, \mathbf{x}, \mathbf{y}) \in \boldsymbol{R}^m \quad \text{into} \quad M \cdot F'(p, \mathbf{x}, \mathbf{y}) = \|F'(p, \mathbf{x}, \mathbf{y})\|_2 \cdot \mathbf{e}_0$$

followed by the solution of

$$\|F'(p, \mathbf{x}, \mathbf{y})\|_2 \cdot \mathbf{e}_0 \cdot \Delta p \approx -M \cdot F(p, \mathbf{x}, \mathbf{y})$$

yielding

$$\Delta p = -\frac{[M \cdot F(p, \mathbf{x}, \mathbf{y})]_0}{\|F'(p, \mathbf{x}, \mathbf{y})\|_2} .$$

See module LinearRegression_I for derivation.

Givens Rotation

Implementation

```
template<typename T, int M>
T Givens(Matrix<T,M,1> a, Matrix<T,M,1> y) {
  auto m=a.size();
  for (auto i=m-2;i>=0;i--) {
    T norm_a_tilde=a.block(i,0,2,1).norm();
    y(i)=(a(i)*y(i)+a(i+1)*y(i+1))/norm_a_tilde;
    y(i+1)=(-a(i+1)*y(i)+a(i)*y(i+1))/norm_a_tilde;
    a(i)=norm_a_tilde; a(i+1)=0;
  }
  return y(0)/a(0);
}

template<typename T, int M>
T Regression(const Matrix<T,M,1> &x, const Matrix<T,M,1> &y, const T& eps) {
  T p=1;
  while (fabs(dEdp(p,x,y))>eps) {
    Matrix<T,M,1> r=F(p,x,y), drdp=dFdp(p,x);
    p-=Givens(drdp,r);
  }
  return p;
}
```

Outline

Solution of the linear regression problem

$$F'(p, \mathbf{x}, \mathbf{y}) \cdot \Delta p \approx -F(p, \mathbf{x}, \mathbf{y})$$

by Householder reflection transforms the vector

$$F'(p, \mathbf{x}, \mathbf{y}) \in \boldsymbol{R}^m \quad \text{into} \quad H \cdot F'(p, \mathbf{x}, \mathbf{y}) = \|F'(p, \mathbf{x}, \mathbf{y})\|_2 \cdot \mathbf{e}_0$$

followed by the solution of

$$\|F'(p, \mathbf{x}, \mathbf{y})\|_2 \cdot \mathbf{e}_0 \cdot \Delta p \approx -H \cdot F(p, \mathbf{x}, \mathbf{y})$$

yielding

$$\Delta p = -\frac{[H \cdot F(p, \mathbf{x}, \mathbf{y})]_0}{\|F'(p, \mathbf{x}, \mathbf{y})\|_2} .$$

See module LinearRegression_I for derivation.

```
template<typename T, int M>
T Householder(Matrix<T,M,1> drdp, Matrix<T,M,1> r) {
  using VT=Matrix<T,M,1>;
  auto m=drdp.size();
  VT v=drdp+drdp(0)/fabs(drdp(0))*drdp.norm()*VT::Unit(m,0);
  drdp-=2*v.dot(drdp)/v.dot(v)*v;
  r-=2*v.dot(r)/v.dot(v)*v;
  return r(0)/drdp(0);
}

template<typename T, int M>
T Regression(const Matrix<T,M,1> &x, const Matrix<T,M,1> &y, const T& eps) {
  T p=1;
  while (fabs(dEdp(p,x,y))>eps) {
    Matrix<T,M,1> r=F(p,x,y), drdp=dFdp(p,x);
    p-=Householder(drdp,r);
  }
  return p;
}
```

Outline

Summary

- Nonlinear regression methods for univariate scalar models based on linearization and
  - normal equation
  - Givens rotation
  - Householder reflection

Next Steps

- Play with sample code.
- Compare results with those obtained by convex minimization methods.
- Continue the course to find out more ...