Iterative Methods for Linear Systems of Equations

Projection methods (1)

ITMAN PhD-course, DTU, 20-10-08 till 24-10-08

Martin van Gijzen

Overview day 2

- The Krylov subspace
  - Better than Richardson
  - Construction of a basis
  - Computation of an approximate solution
- Solution methods:
  - The Conjugate Gradient method
  - CG Convergence theory
  - CR and MINRES
- CG for the normal equations

Richardson’s method

Yesterday we saw that Richardson’s method produces iterates

$$x_{k+1} = x_0 + \sum_{i=0}^{k} (I - A)^i r_0 \ .$$

So $x_{k+1} \in span\{x_0, r_0, Ar_0, A^2 r_0, \cdots, A^k r_0\}$.
For the residuals we have

$$r_{k+1} = b - Ax_0 - A\sum_{i=0}^{k} (I - A)^i r_0 = r_0 - A\sum_{i=0}^{k} (I - A)^i r_0 \ .$$

and hence $r_{k+1} \in span\{r_0, Ar_0, A^2 r_0, \cdots, A^{k+1} r_0\}$.

The Krylov subspace

The space $span\{r_0, Ar_0, A^2r_0, \cdots, A^{k-1}r_0\}$ is called the *Krylov subspace* of dimension k, corresponding to matrix A and initial residual r_0 and is denoted by

$$K^k(A; r_0) = span\{r_0, Ar_0, A^2r_0, \cdots, A^{k-1}r_0\}$$

The leading question in the next days will be:
Can we use the information contained in $K^k(A; r_0)$ more efficiently than in Richardson's method?

Basic Iterative Methods

Basic iterative methods produce iterates according to

$$x_{k+1} = x_k + M^{-1} r_k \ .$$

Hence $x_{k+1} \in x_0 \cup K^{k+1}(M^{-1}A; M^{-1}r_0)$.

For the moment, we concentrate on the choice $M = I$.

One-step methods are Krylov methods

Both the steepest-descent method and the minimal-residual method perform iterations of the form

$$\begin{aligned} x_{k+1} &= x_k + \alpha_k r_k \\ r_{k+1} &= b - Ax_{k+1} = r_k - \alpha_k A r_k \ . \end{aligned} \tag{1}$$

Clearly, also for these methods $x_{k+1} \in x_0 \cup K^{k+1}(A; r_0)$

Towards optimal methods

One-step projection methods make an optimal combination of the last two basis vectors in the Krylov subspace. The question that we will address now is:

Is it possible to make optimal linear combinations of all the basis vectors in the Krylov subspace?

We will answer this question in two steps:

- First, we discuss how to construct a basis for $K^k(A; r_0)$;
- Then, we will explain how we can construct an optimal approximation as a linear combination of the basis vectors. Today we will focus on symmetric matrices.

A basis for the Krylov subspace

The most simple basis for the Krylov subspace $K^k(A; r_0)$ is the power basis: $r_0, Ar_0, A^2r_0, \cdots A^{k-1}r_0$.

Clearly, this basis is ill-conditioned since $A^{k-1}r_0$ will point more and more in the direction of the largest eigenvector of A.

A stable, orthogonal basis can be constructed with Arnoldi's method.

Arnoldi’s method

Choose a starting vector q_1 with $\|q_1\|_2 = 1$.

```
FOR   $k = 1, \cdots$ DO               iteration
      $v = Aq_k$                       expansion
      FOR $i = 1, ..., k$              orthogonalisation
          $h_{i,k} = v^T q_i$
          $v = v - h_{i,k} q_i$
      END FOR
      $h_{k+1,k} = \|v\|_2$
      IF $h_{k+1,k} = 0$ STOP          invariant subspace spanned
      $q_{k+1} = v / h_{k+1,k}$        new basis vector
END   FOR
```

The Arnoldi relation

The Arnoldi method can be summarised in a compact way. Let

$$H_k = \begin{bmatrix} h_{1,1} & \cdots & \cdots & h_{1,k} \\ h_{2,1} & \ddots & & \vdots \\ & \ddots & \ddots & \vdots \\ O & & h_{k,k-1} & h_{k,k} \end{bmatrix}$$

and $Q_k = [q_1 \; q_2 \cdots q_k]$ then

$$AQ_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T$$

Here e_k is the $k-th$ canonical basis vector in $\mathbb{R}^k$.

A is symmetric

According to the Arnoldi relation we have

$$Q_k^T A Q_k = H_k \ .$$

Moreover, if A is symmetric we have

$$H_k^T = Q_k^T A^T Q_k = Q_k^T A Q_k = H_k \ .$$

So H_k is symmetric and upper Hessenberg,

this means that H_k must be tridiagonal.

A is symmetric (2)

So

$$H_k = \begin{bmatrix} h_{1,1} & h_{2,1} & & O \\ h_{2,1} & \ddots & \ddots & \\ & \ddots & \ddots & h_{k,k-1} \\ O & & h_{k,k-1} & h_{k,k} \end{bmatrix} .$$

With $\alpha_k = h_{k,k}$ and $\beta_k = h_{k-1,k}$ the Arnoldi method simplifies to the (symmetric) Lanczos method. With the Lanczos method it is possible to compute a new orthonormal basis vector using only the two previous basis vectors.

The Lanczos method

Choose a starting vector q_1 with $\|q_1\|_2 = 1$

```
  $\beta_1 = 0$      $q_0 = 0$                                      initialization
  FOR                $k = 1, \cdots$ DO                             iteration
                     $\alpha_k = q_k^T A q_k$
                     $v = Aq_k - \alpha_k q_k - \beta_k q_{k-1}$    new direction
                                                                    orthogonal to
                     $\beta_{k+1} = \|v\|_2$                        previous $q$
                     $q_{k+1} = v/\beta_{k+1}$                      normalization
  END FOR
```

The Lanczos method

Let

$$T_k = \begin{bmatrix} \alpha_1 & \beta_2 & & & 0 \\ \beta_2 & \alpha_2 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & 0 & \ddots & \ddots & \beta_k \\ & & & \beta_k & \alpha_k \end{bmatrix}.$$

and

$$Q_k = [q_1 \; q_2 \ldots q_k]$$

Then $AQ_k = Q_kT_k + \beta_{k+1}q_{k+1}e_k^T$

Eigenvalue methods

Arnoldi’s and Lanczos method were originally proposed as iterative methods to compute the eigenvalues of a matrix A:

$$Q_k^T A Q_k = H_k$$

is ’almost’ a similarity transformation. The eigenvalues of H_k are called Ritz values.

Optimal approximations (1)

The Lanczos method provides a cheap way to compute an orthogonal basis for the Krylov subspace $K^k(A; r_0)$. Our approximations can be written as

$$x_k = x_0 + Q_k y_k$$

where y_k is determined so that either the error

$$f(x_k) = \|x_k - x\|_A^2 = (x_k - x)^T A (x_k - x)$$

is minimised in A-norm (only meaningful if A is pos. def.) or that

$$g(x_k) = \|A(x_k - x)\|_2^2 = r_k^T r_k,$$

i.e. the norm of the residual is minimised.

Optimal approximations (2)

We first look at minimisation of the error in the A-norm:

$$x_k = x_0 + Q_k y_k \Rightarrow f(x_k) = (x_0 + Q_k y_k - x)^T A (x_0 + Q_k y_k - x) \ .$$

To minimise, we differentiate with respect to y_k and set $\frac{\partial f(x_k)}{\partial y_k} = 0$. This yields

$$Q_k^T A Q_k y_k = Q_k^T r_0$$

and with $Q_k^T A Q_k = T_k, r_0 = \|r_0\| q_1$ we get

$$T_k y_k = \|r_0\| e_1$$

with e_1 the first canonical basis vector.

Optimal approximations (3)

It is easy to see that the residuals are orthogonal to the basis vectors:

$$r_k = r_0 - AQ_k y_k \Rightarrow Q_k^T r_k = Q_k^T r_0 - Q_k^T AQ_k y_k = 0$$

This condition is equivalent to minimising $f(x_k)$ when A is SPD.

Since the r_k's are orthogonal, each residual is just a multiple of the corresponding basis vector q_k. This also means that the residuals form an orthogonal basis for the Krylov subspace.

Towards a practical algorithm

The main problem in the Lanczos algorithm is that all q_i (or r_i) have to be stored to compute x_k. This problem can be overcome by making an implicit LU factorisation of T_k and updating x_k in every iteration.

Details can be found in the books of Van der Vorst and of Saad.

With this technique we get the famous and very simple Conjugate Gradient method.

The Conjugate Gradient method

```
$r_0 = b - Ax_0; \quad p_0 = r_0$                initialization
FOR    $k = 0, 1, \cdots,$  DO
       $\alpha_k = \frac{r_k^T r_k}{p_k^T A p_k}$
       $x_{k+1} = x_k + \alpha_k p_k$            update iterate
       $r_{k+1} = r_k - \alpha_k A p_k$          update residual
       $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
       $p_{k+1} = r_{k+1} + \beta_k p_k$         update direction vector
END FOR
```

Properties of CG

CG has several favourable properties:

- The method uses limited memory: only three vectors need to be stored;
- The method is optimal: the error is minimised in A-norm;
- The method is finite: the $n+1$-st residual must be zero since all the residuals are orthogonal;
- The method is robust: $p_k^T A p_k$ and $r_k^T r_k$ both imply that the true solution has been found (that $r_k = 0$).

Lanczos matrix

Since CG and Lanczos are mathematically equivalent it should be possible to recover the Lanczos matrix T_k from the CG-iteration parameters. This is indeed the case

$$T_k = \begin{bmatrix} \frac{1}{\alpha}0 & \frac{\sqrt{\beta_0}}{\alpha_0} & & & 0 \\ \frac{\sqrt{\beta_0}}{\alpha_0} & \frac{1}{\alpha}1 + \frac{\beta_0}{\alpha_0} & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & 0 & \ddots & \ddots & \frac{\sqrt{\beta_{m-2}}}{\alpha_{m-2}} \\ & & & \frac{\sqrt{\beta_{k-2}}}{\alpha_{k-2}} & \frac{1}{\alpha}k-1 + \frac{\beta_{k-2}}{\alpha_{k-2}} \end{bmatrix}.$$

Conjugate Gradient Error Analysis

CG minimises the function

$$f(x_k) = (x_0 + Q_k y_k - x)^T A (x_0 + Q_k y_k - x)\ .$$

With the error $e_k = x_k - x$ and $Ae_0 = r_0$ CG minimises

$$\|e_k\|_A = \|e_0 + Q_k y_k\| \text{ with } Q_k y_k \in K^k(A; Ae_0)$$

Consequently

$$e_k = e_0 - \alpha_1 A e_0 - \alpha_2 A^2 e_0 - \cdots - \alpha_k A^k e_0 = p_k(A) e_0$$

So e_k is a polynomial in A times e_0, with $p_k(0) = 1$:

$$\|e_k\|_A = \min_{p_k} \|p_k(A) e_0\|_A \le \min_{p_n} \max_i |p_k(\lambda_i(A))| \cdot \|e_0\|_A$$

A convergence bound for CG

The iterates x_k obtained from the CG algorithm satisfy the following inequality:

$$\|x - x_k\|_A \leq 2 \left(\frac{\sqrt{K_2(A)} - 1}{\sqrt{K_2(A)} + 1} \right)^k \|x - x_0\|_A .$$

$K_2(A)$ is the condition number of A, which is for SPD-matrices

$$K_2(A) = \frac{\lambda_{max}}{\lambda_{min}}$$

Proof (sketch)

The error $e_k = x - x_k$ can be written as $p_k(A)e_0$ with $p_k(x)$ a polynomial such that $p_k(0) = 1$. Hence we have

$$\|e_k\|_A = \|p_k(A)e_0)\|_A$$

Since CG is optimal, we have

$$\|p_k(A)e_0)\|_A \leq \|q_k(A)e_0\|_A \;\; \forall \, q_k(x) \; with \; q_k(0) = 1$$

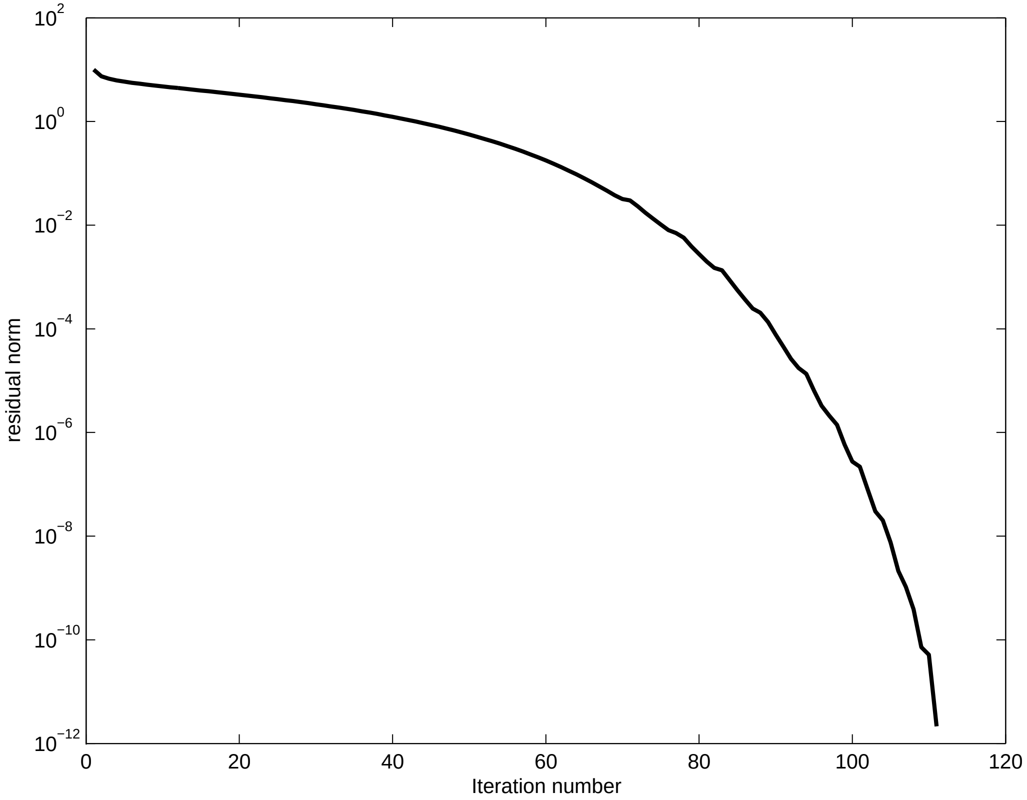
The convergence bound can now be proved by taking for $q_k(x)$ Chebychev polynomials that are transformed to the interval $\lambda_{min}, \lambda_{max}$.

Superlinear convergence

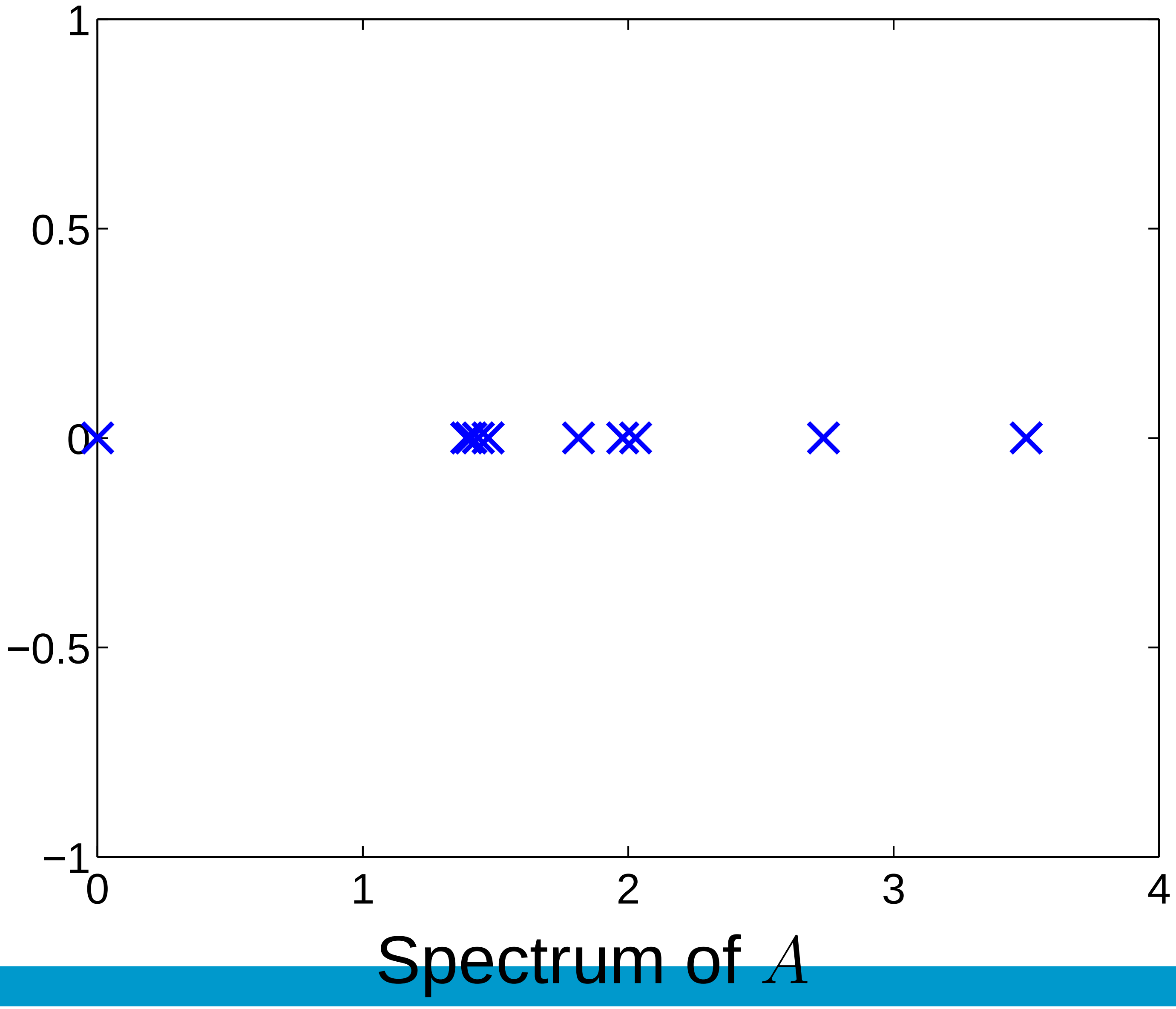
The upperbound on the CG-error is in practice very pessimistic.

Typically the rate of convergence increases during the process.

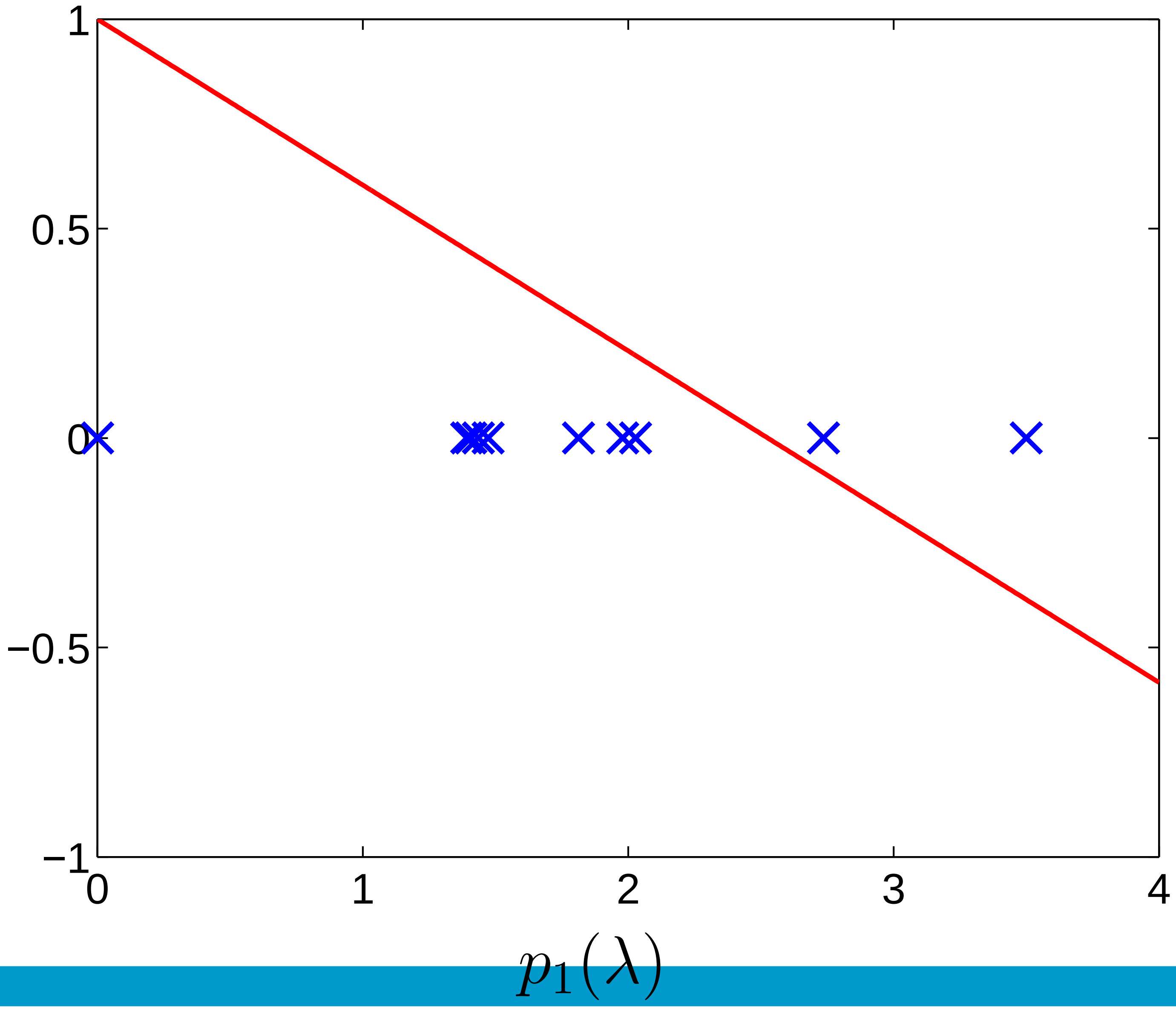
This is called superlinear convergence.



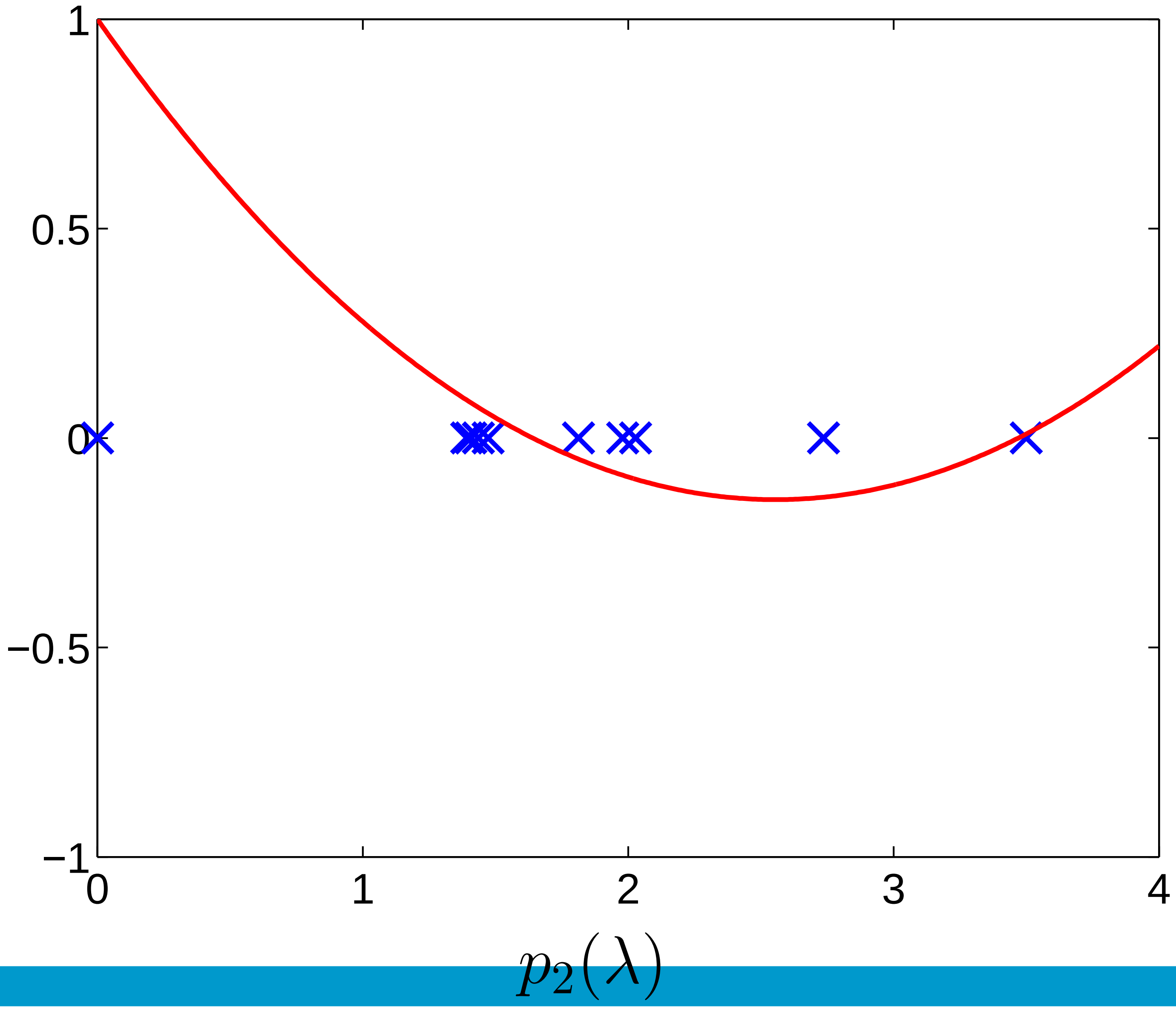
CG convergence: Sparse Spectrum



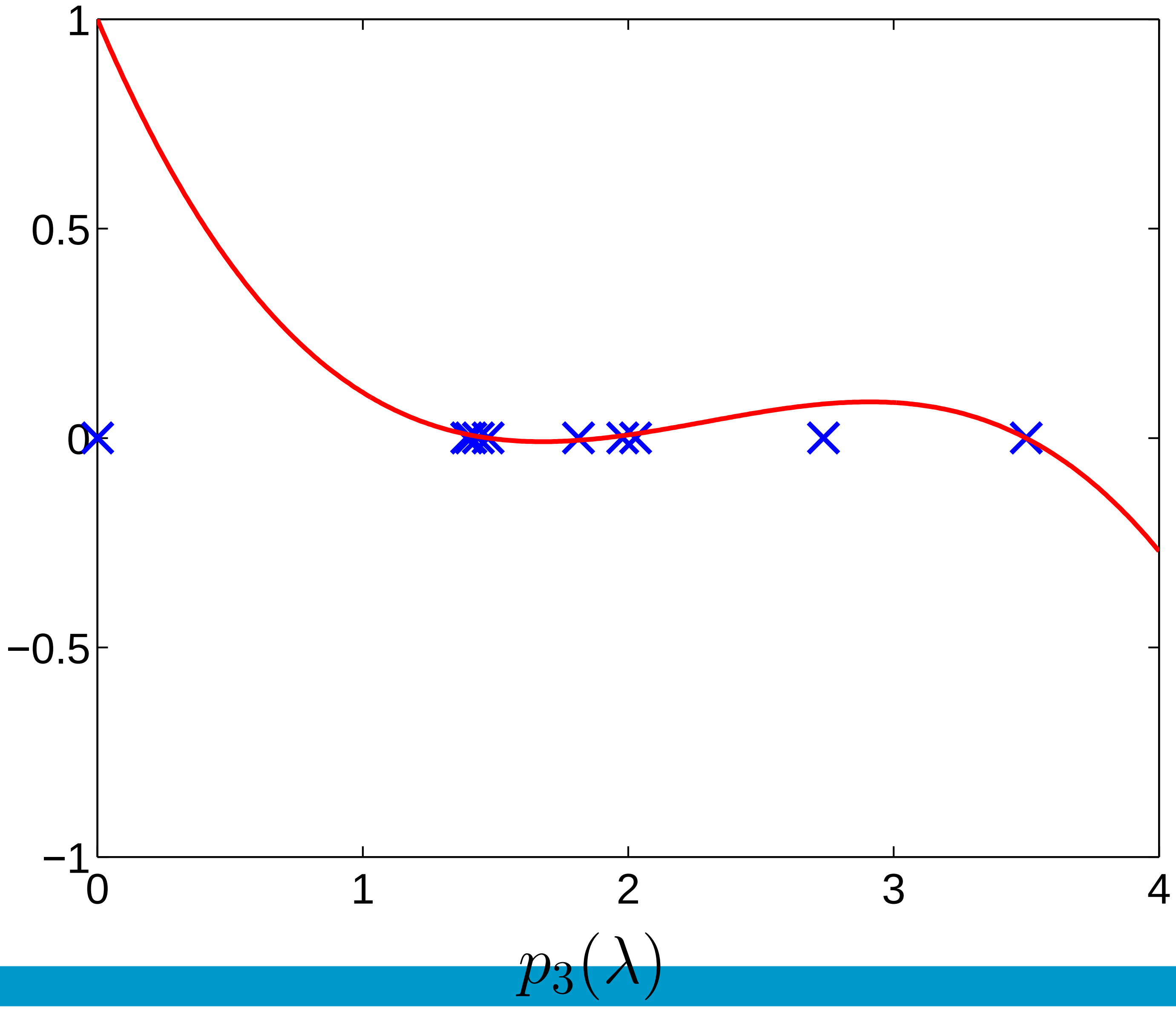
CG convergence: Sparse Spectrum



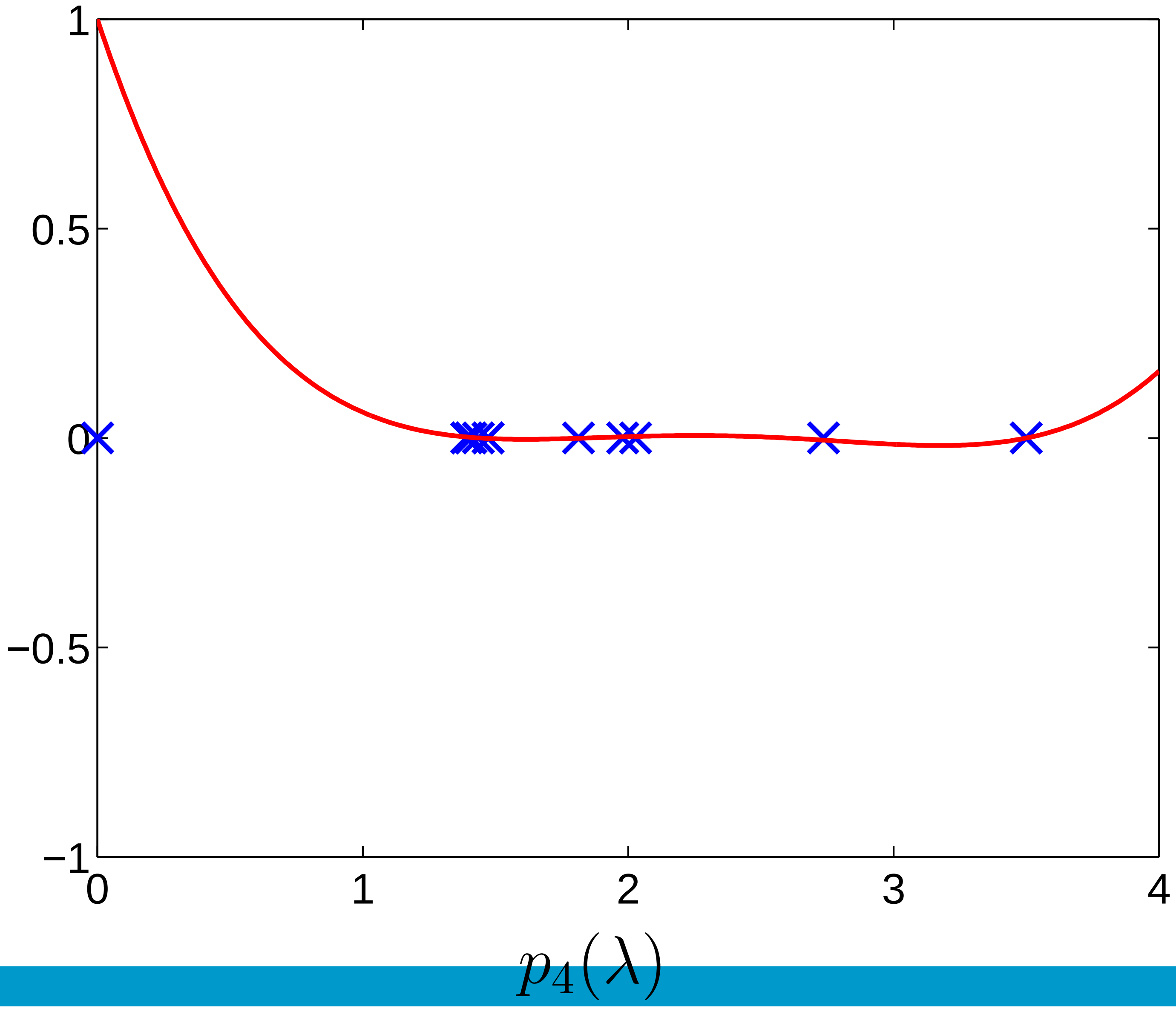
CG convergence: Sparse Spectrum



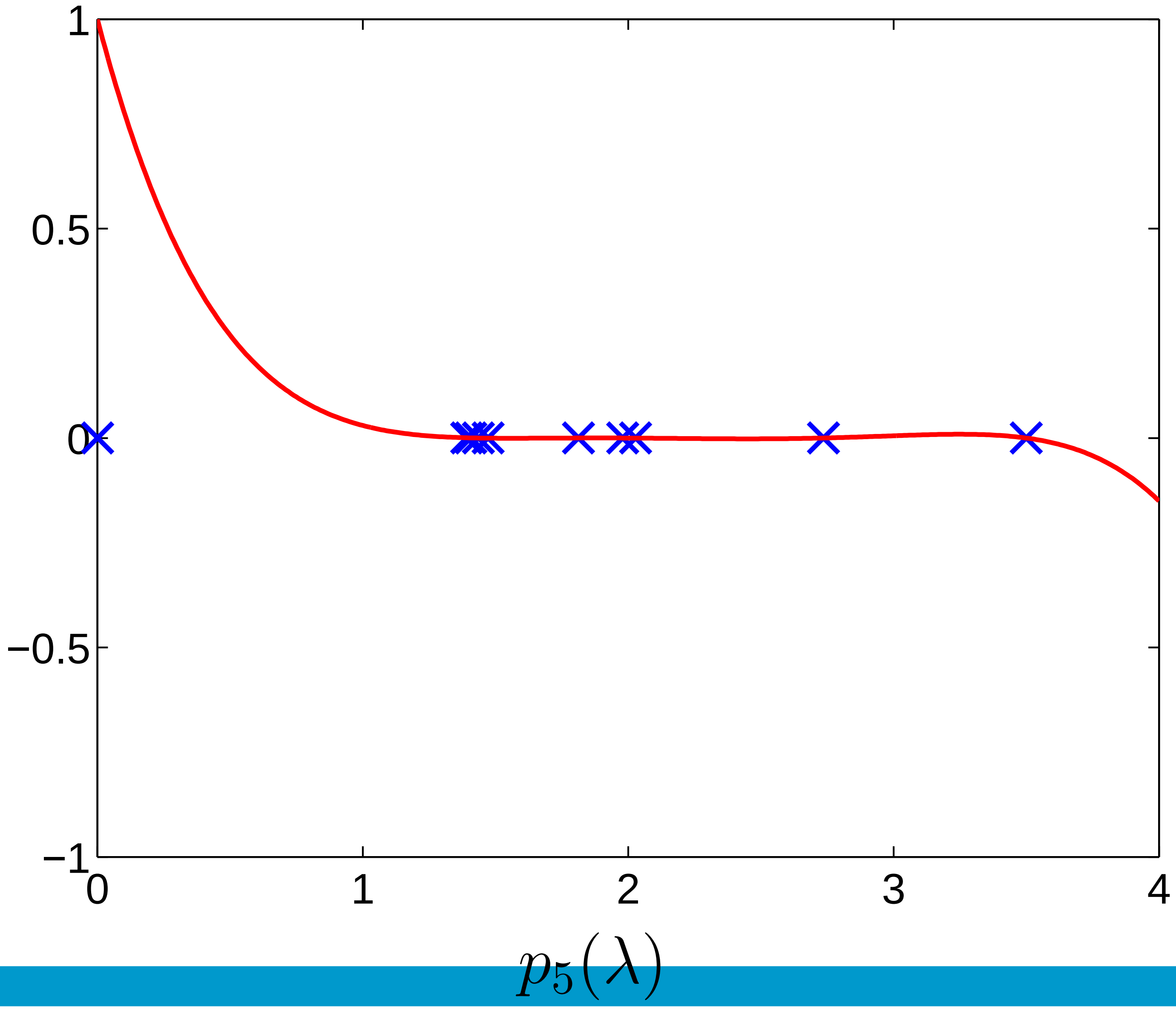
CG convergence: Sparse Spectrum



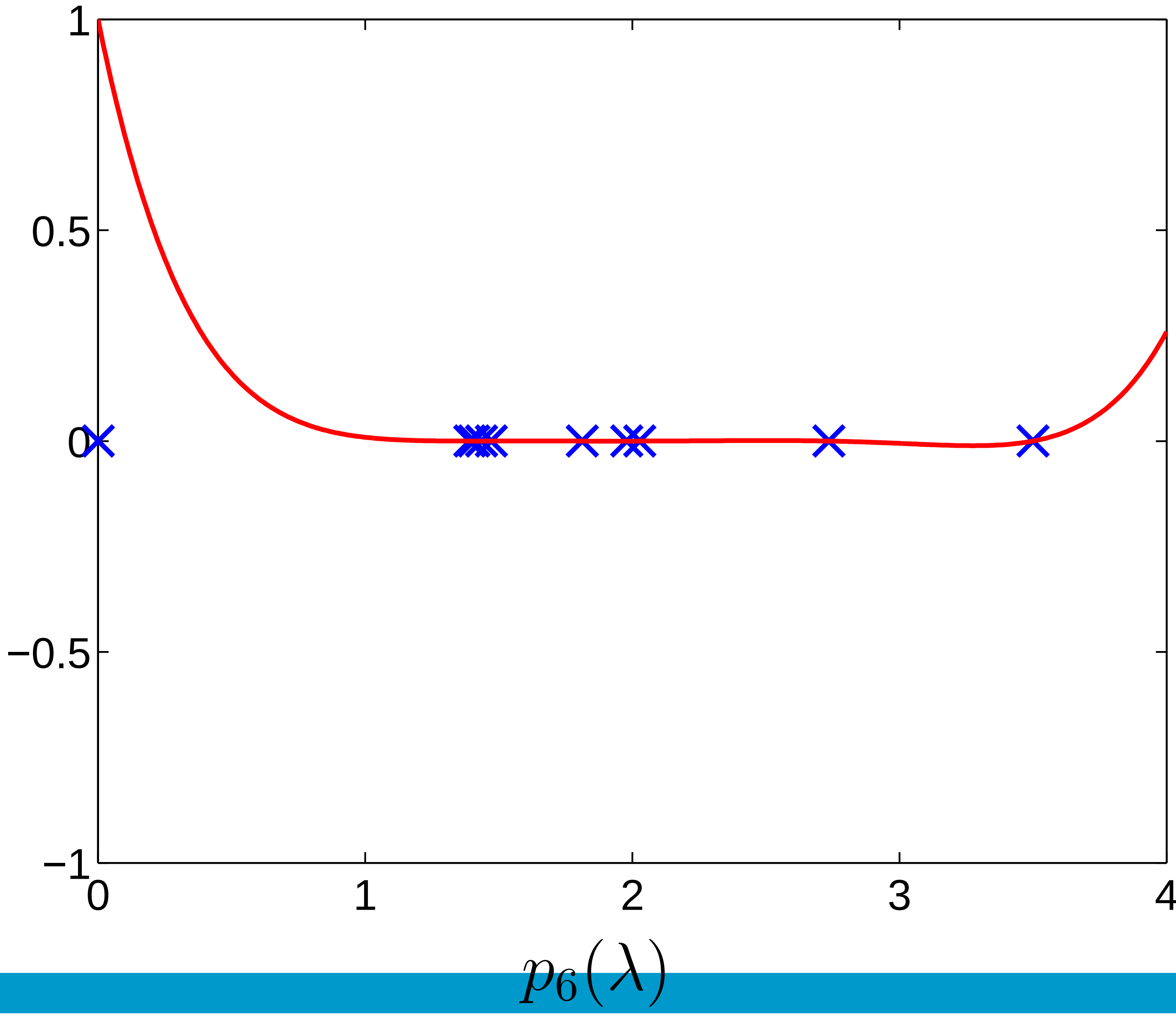
CG convergence: Sparse Spectrum



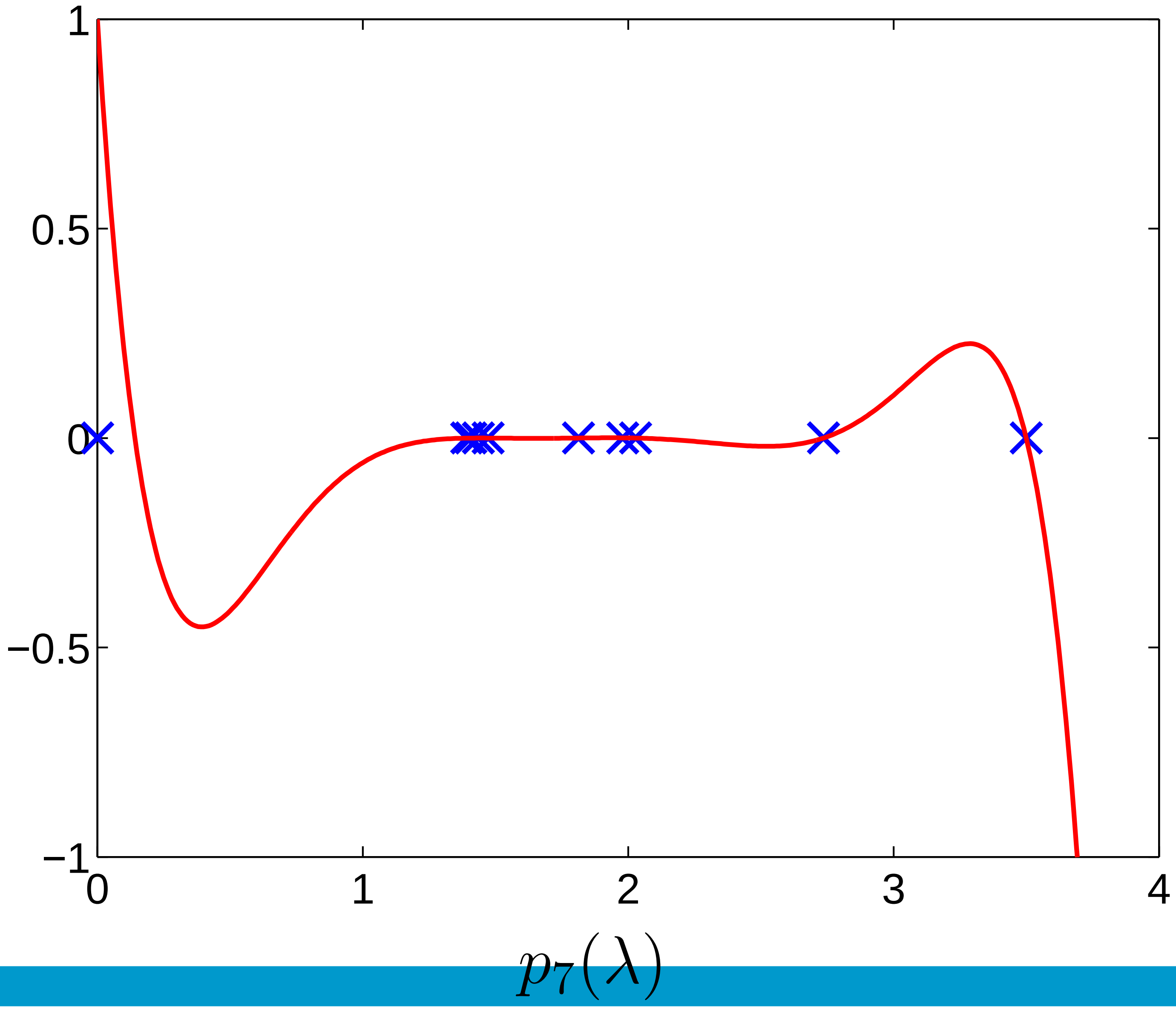
CG convergence: Sparse Spectrum



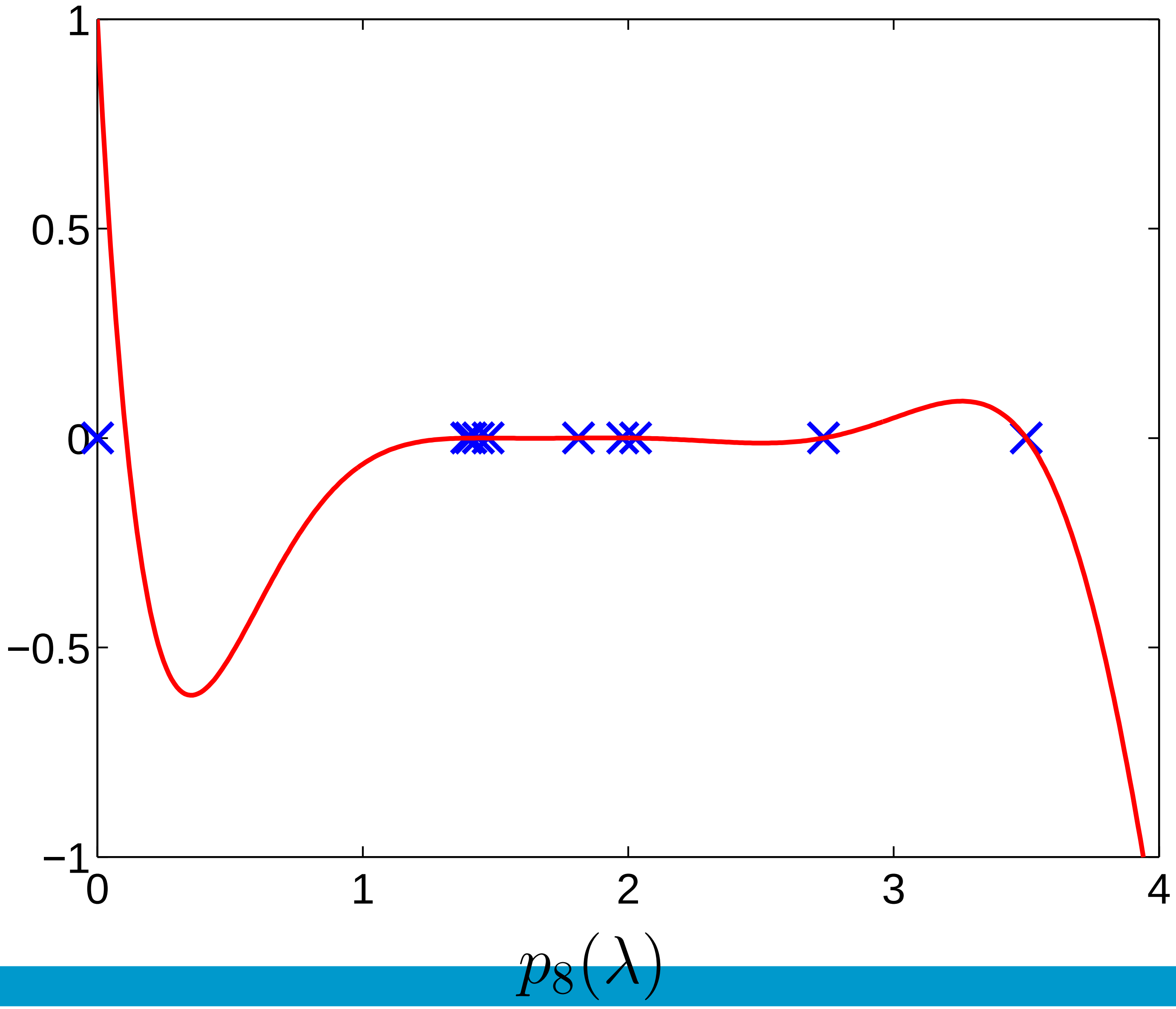
CG convergence: Sparse Spectrum



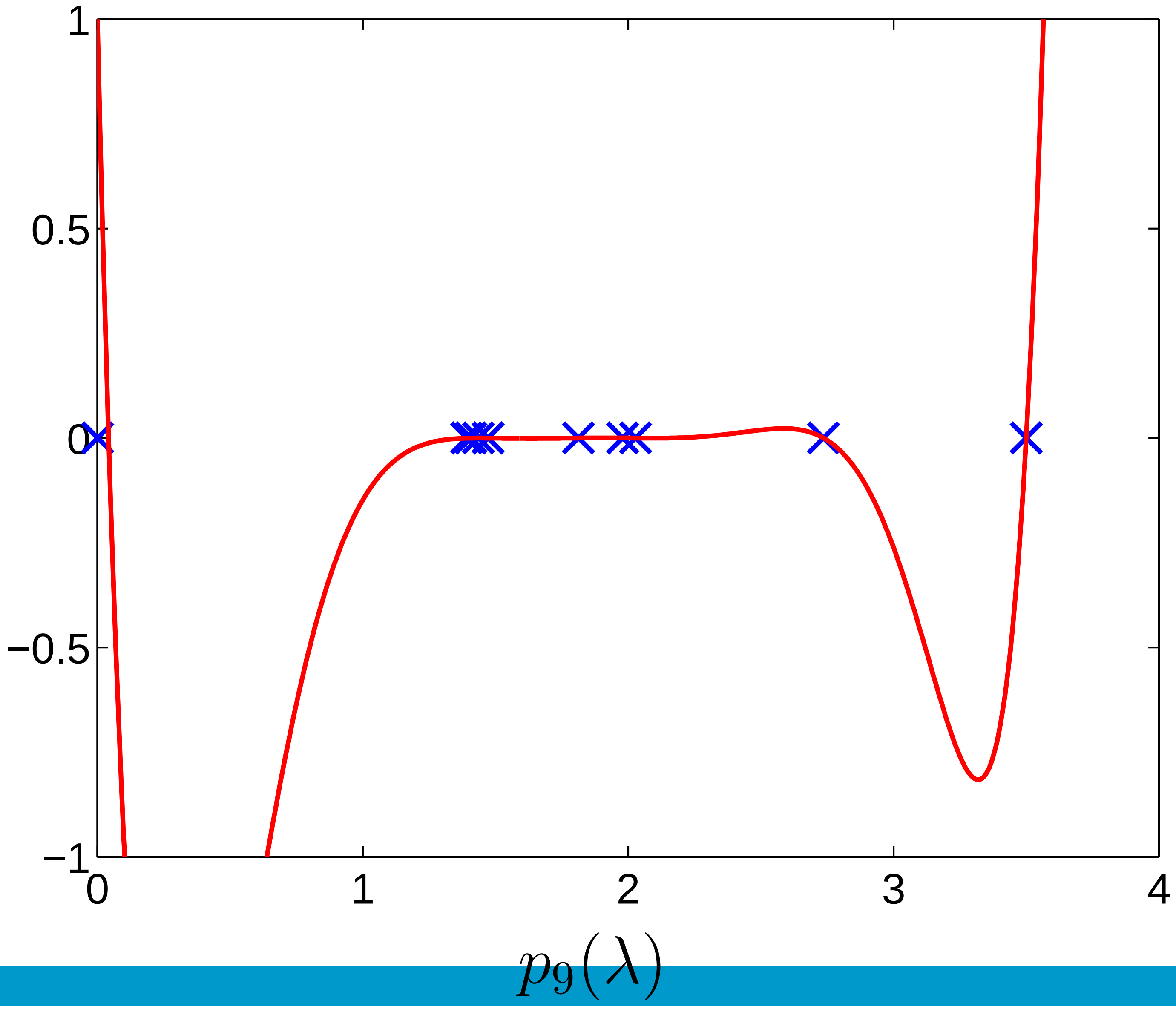
CG convergence: Sparse Spectrum



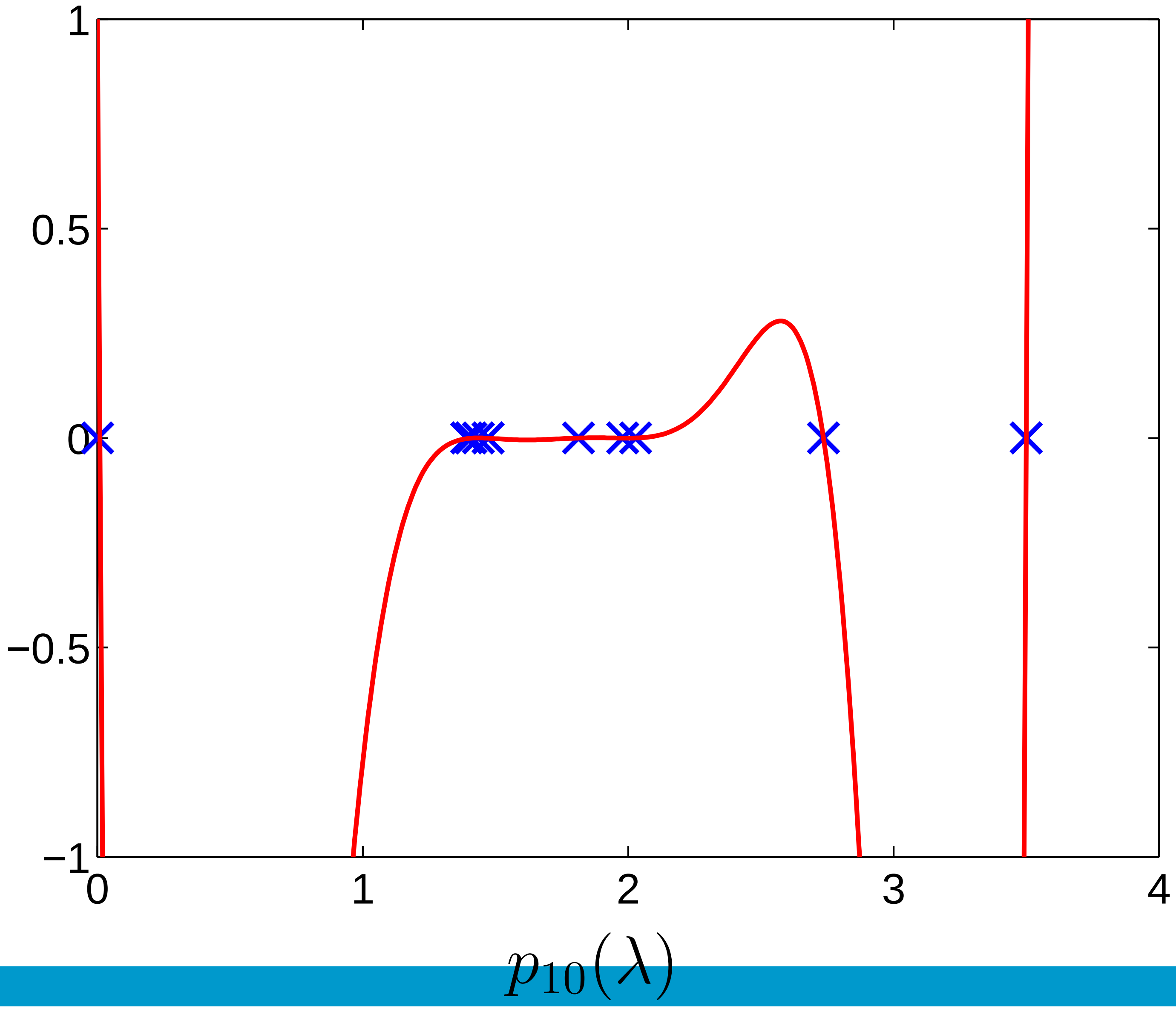
CG convergence: Sparse Spectrum



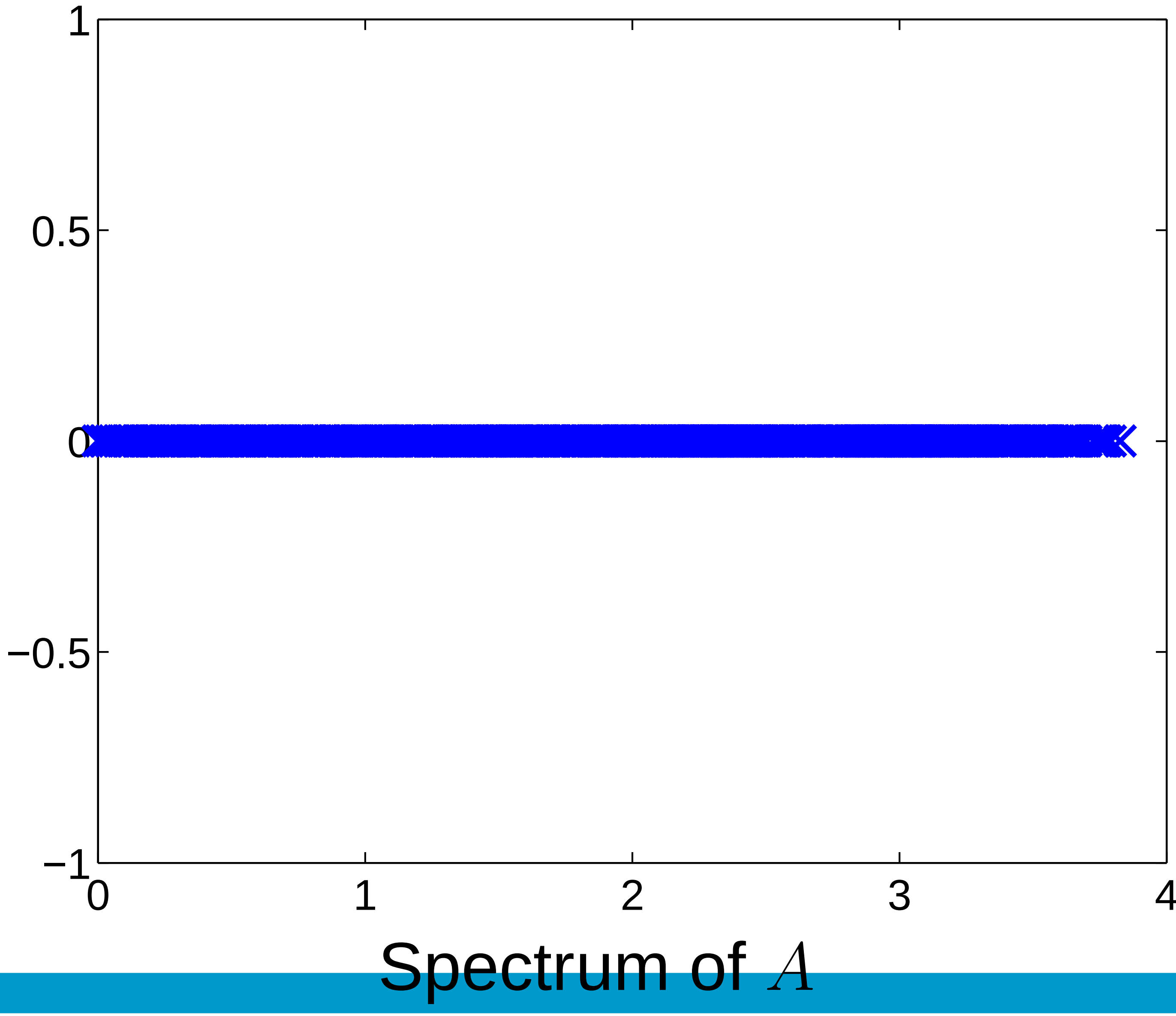
CG convergence: Sparse Spectrum



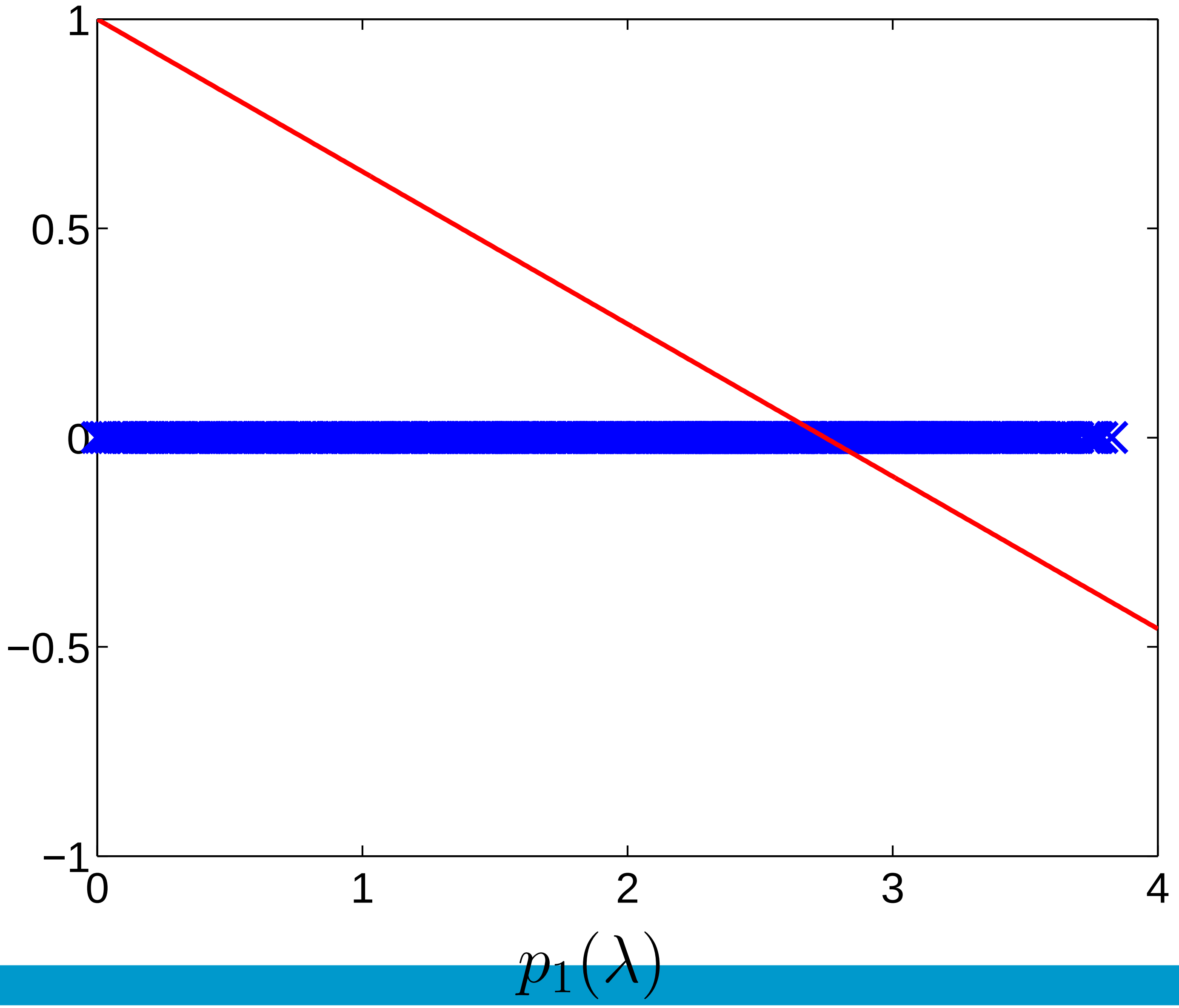
CG convergence: Sparse Spectrum



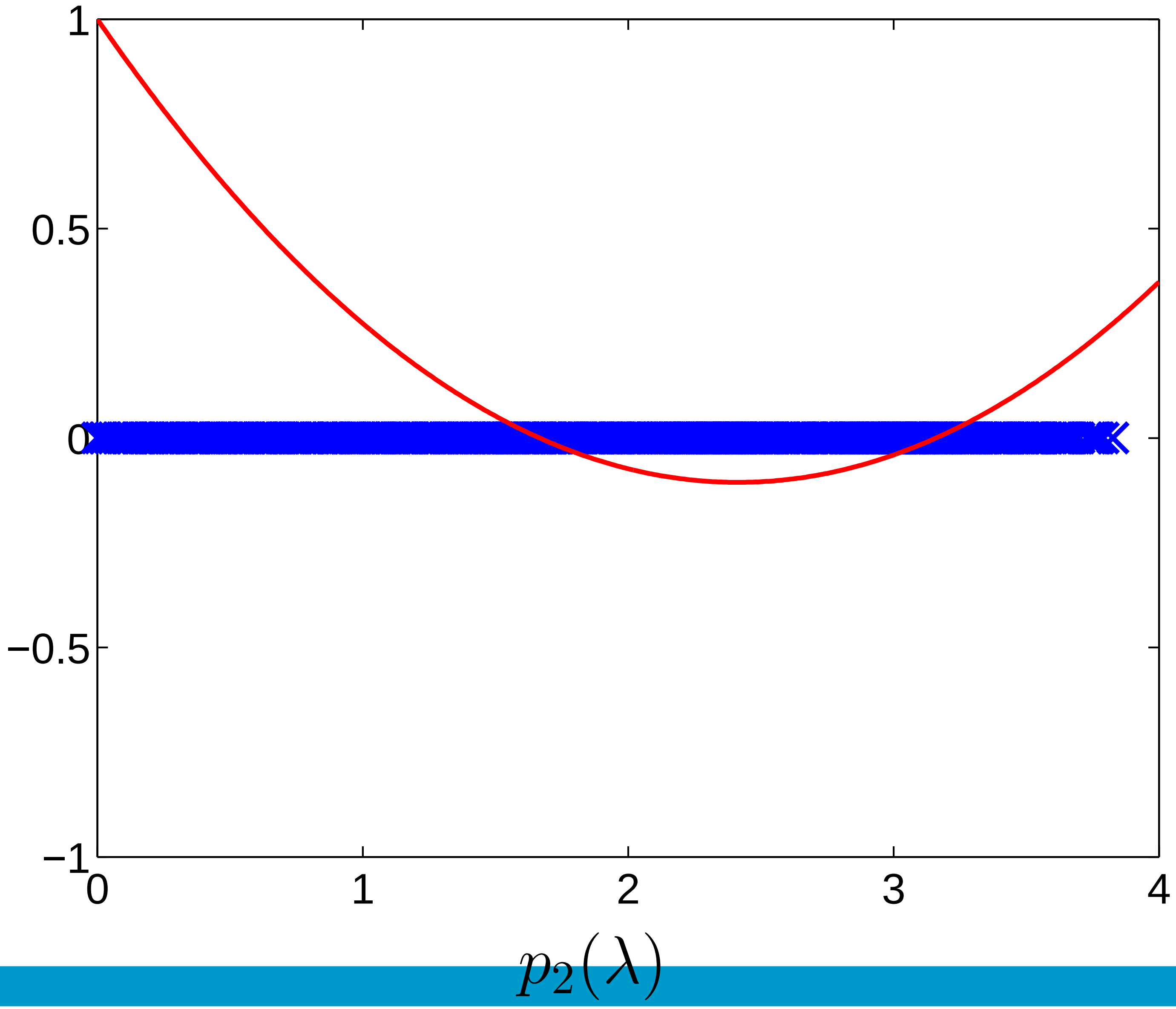
CG convergence: Dense Spectrum



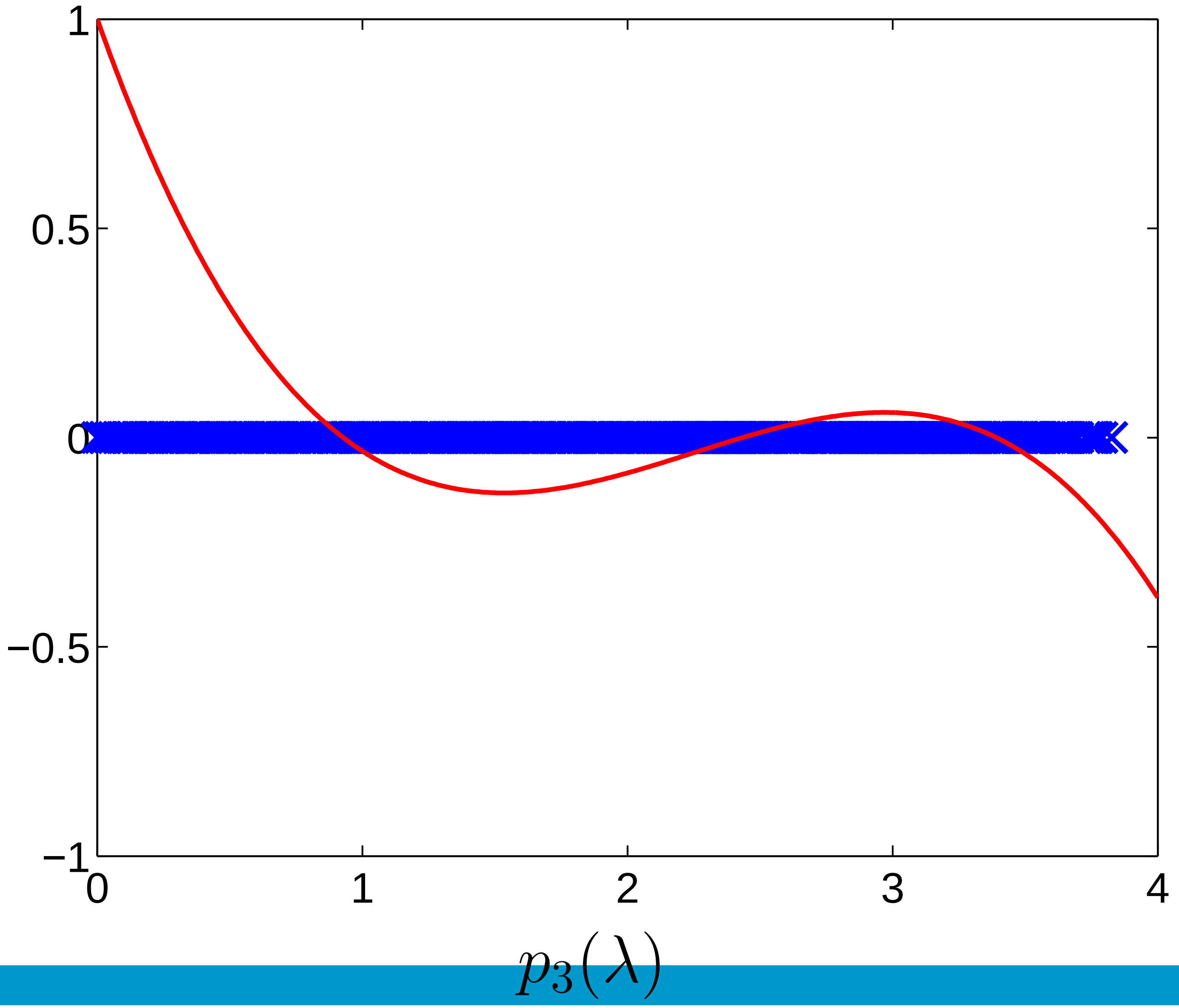
CG convergence: Dense Spectrum



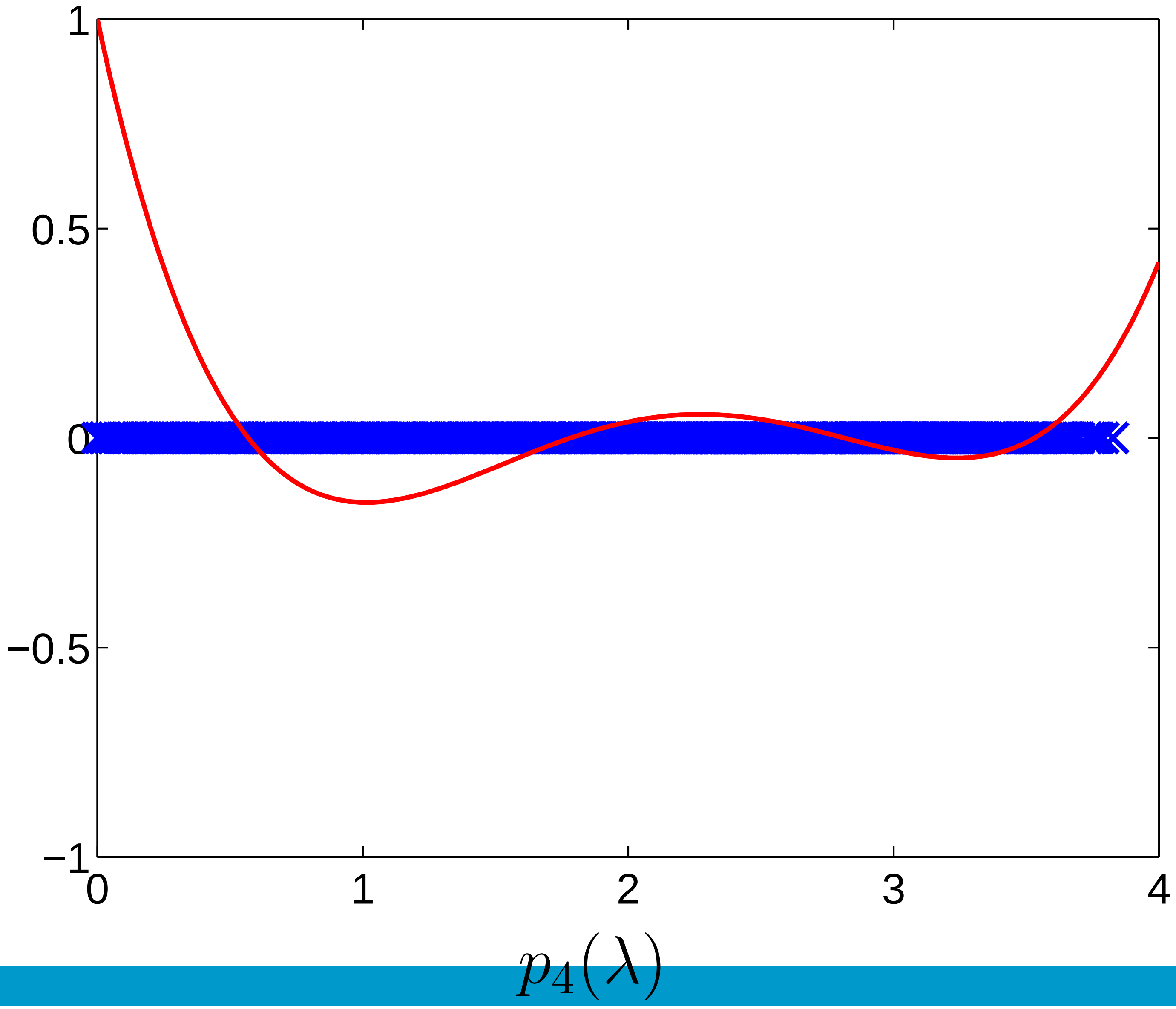
CG convergence: Dense Spectrum



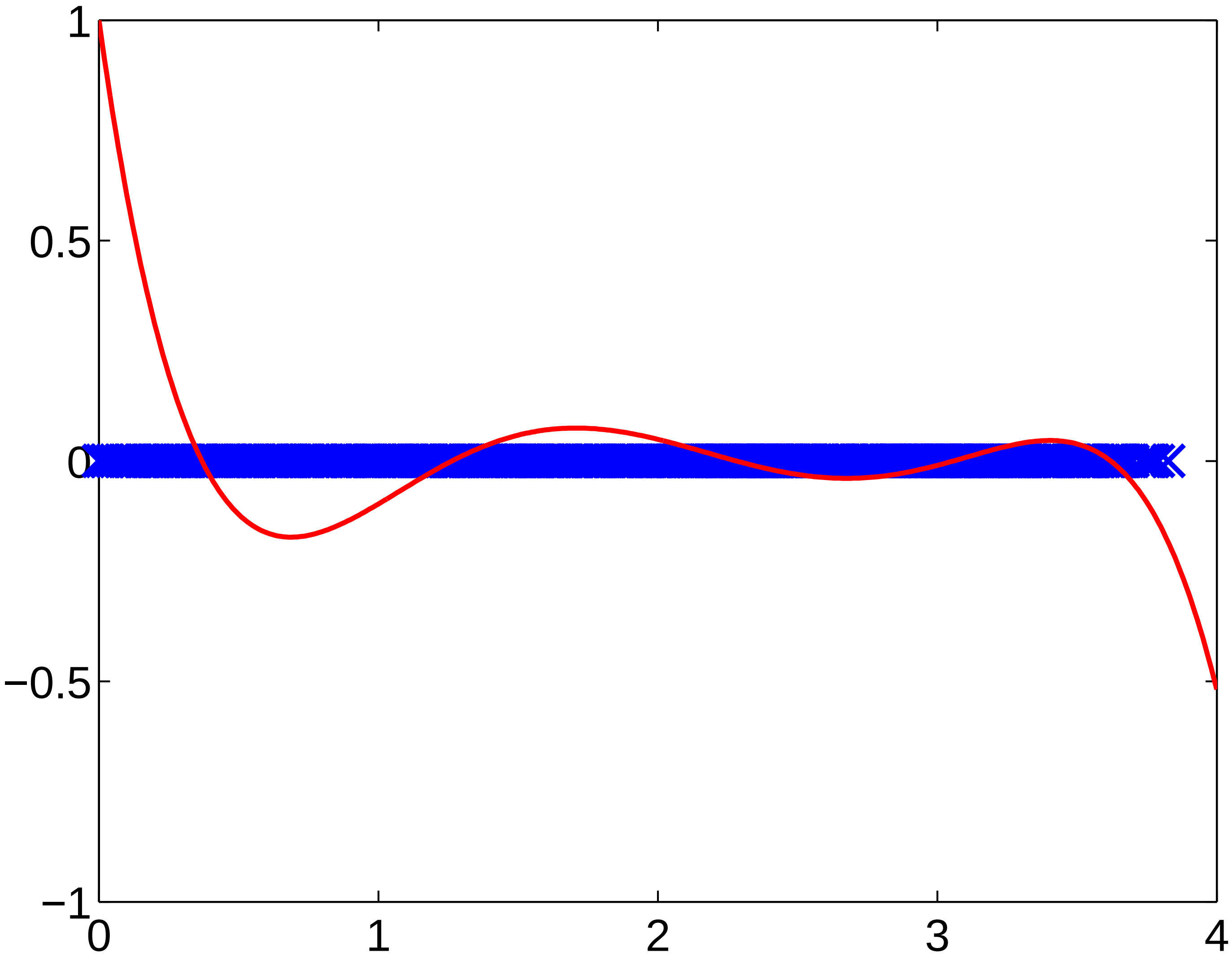
CG convergence: Dense Spectrum



CG convergence: Dense Spectrum

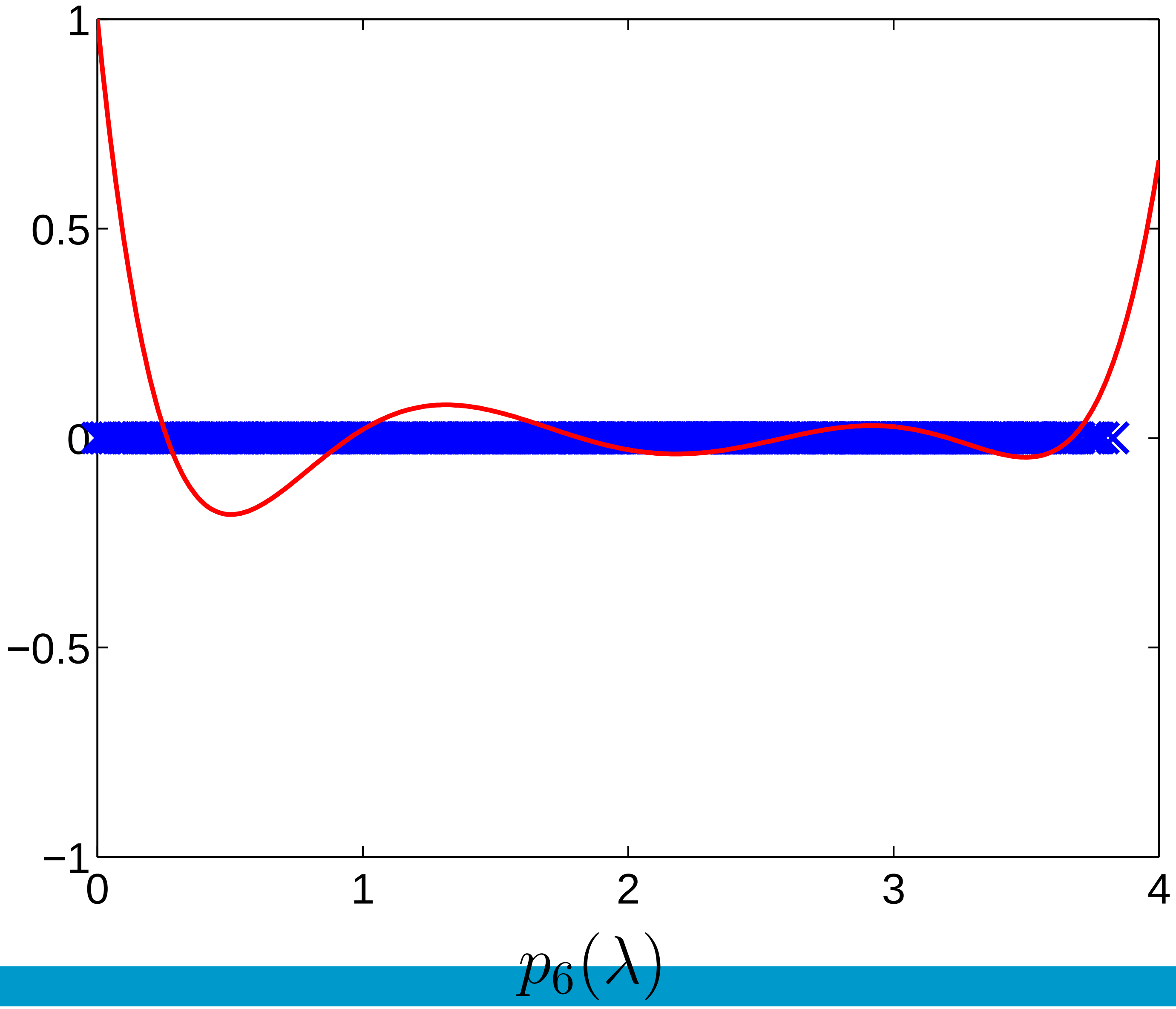


CG convergence: Dense Spectrum

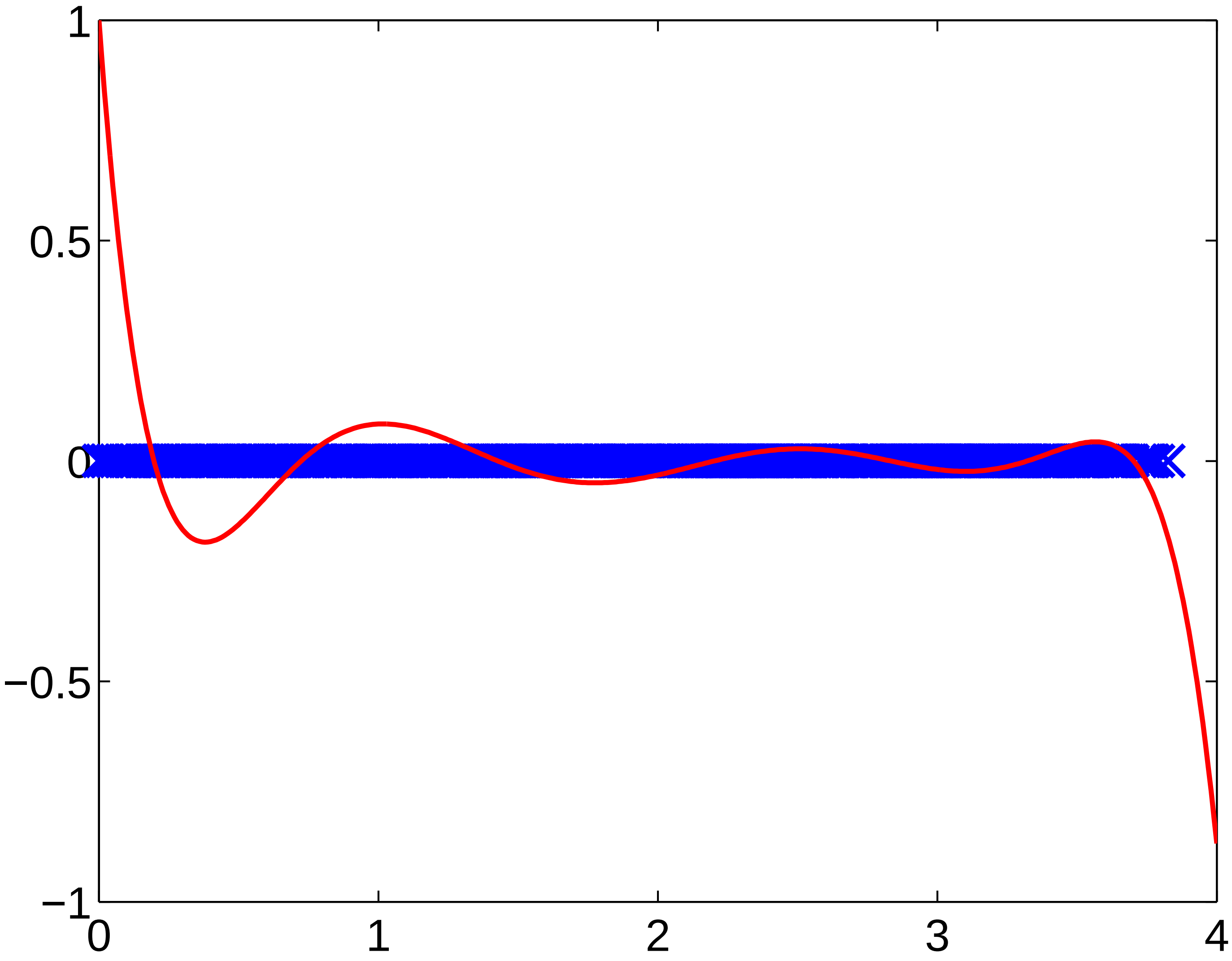


$p_5(\lambda)$

CG convergence: Dense Spectrum

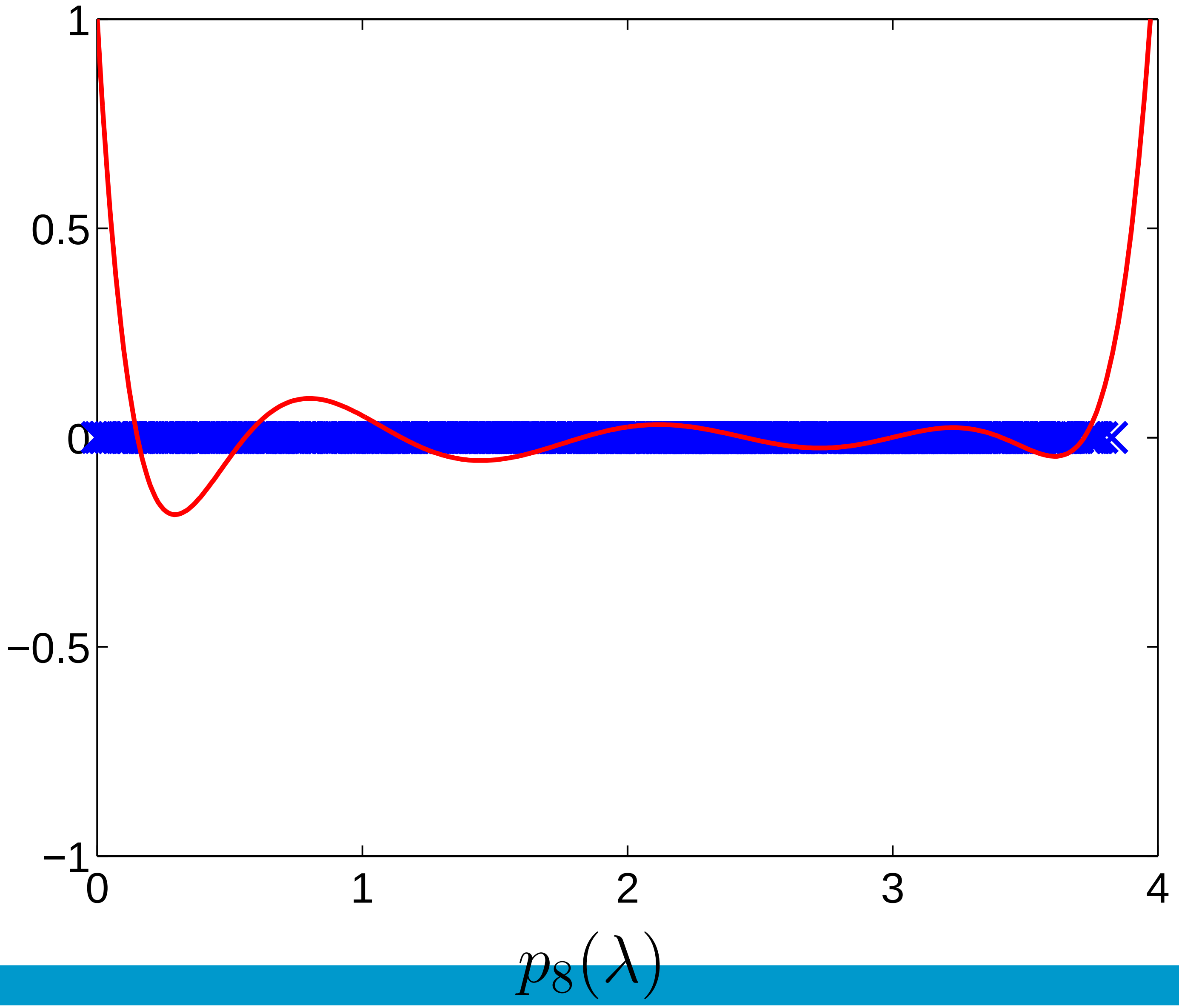


CG convergence: Dense Spectrum

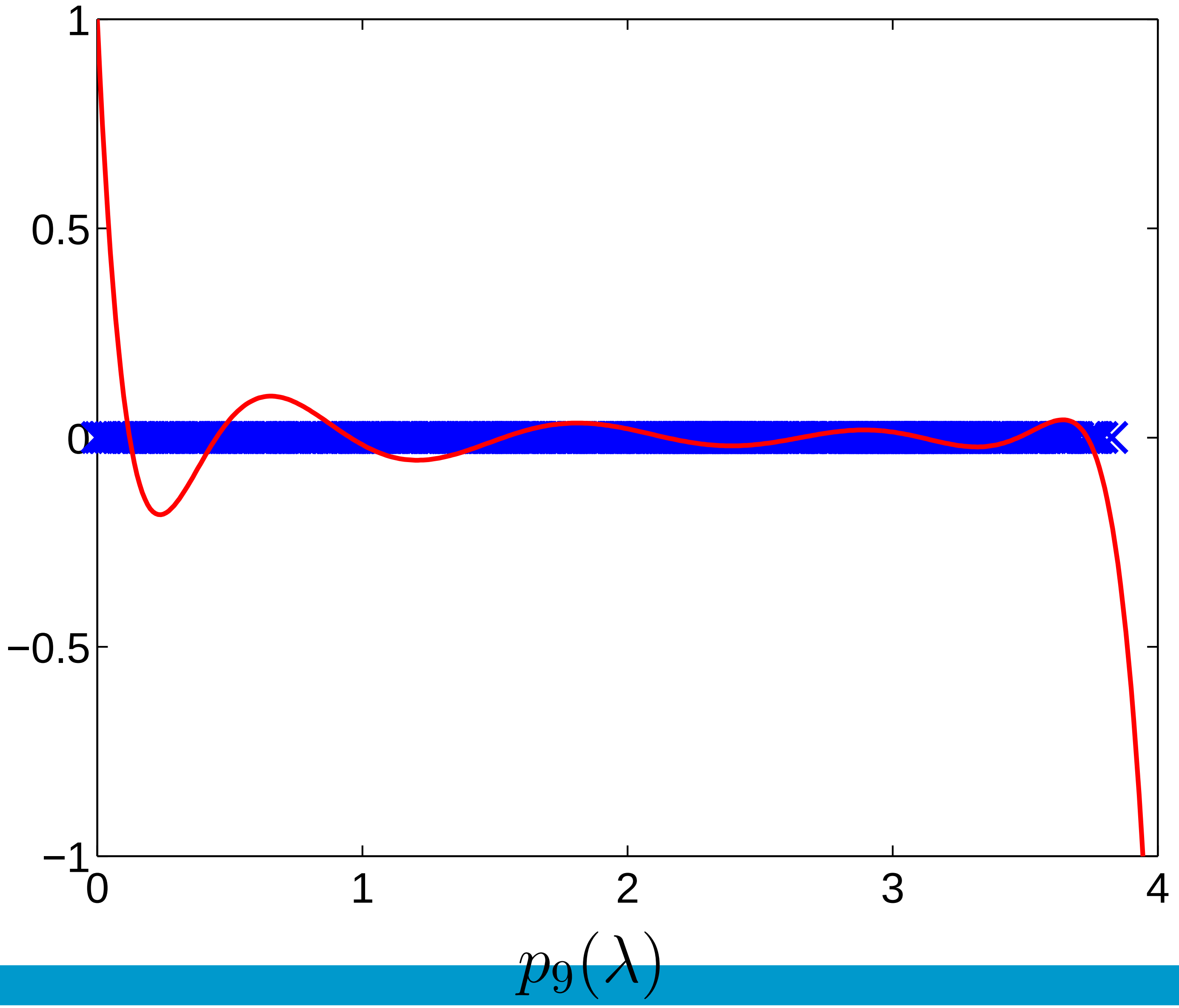


$p_7(\lambda)$

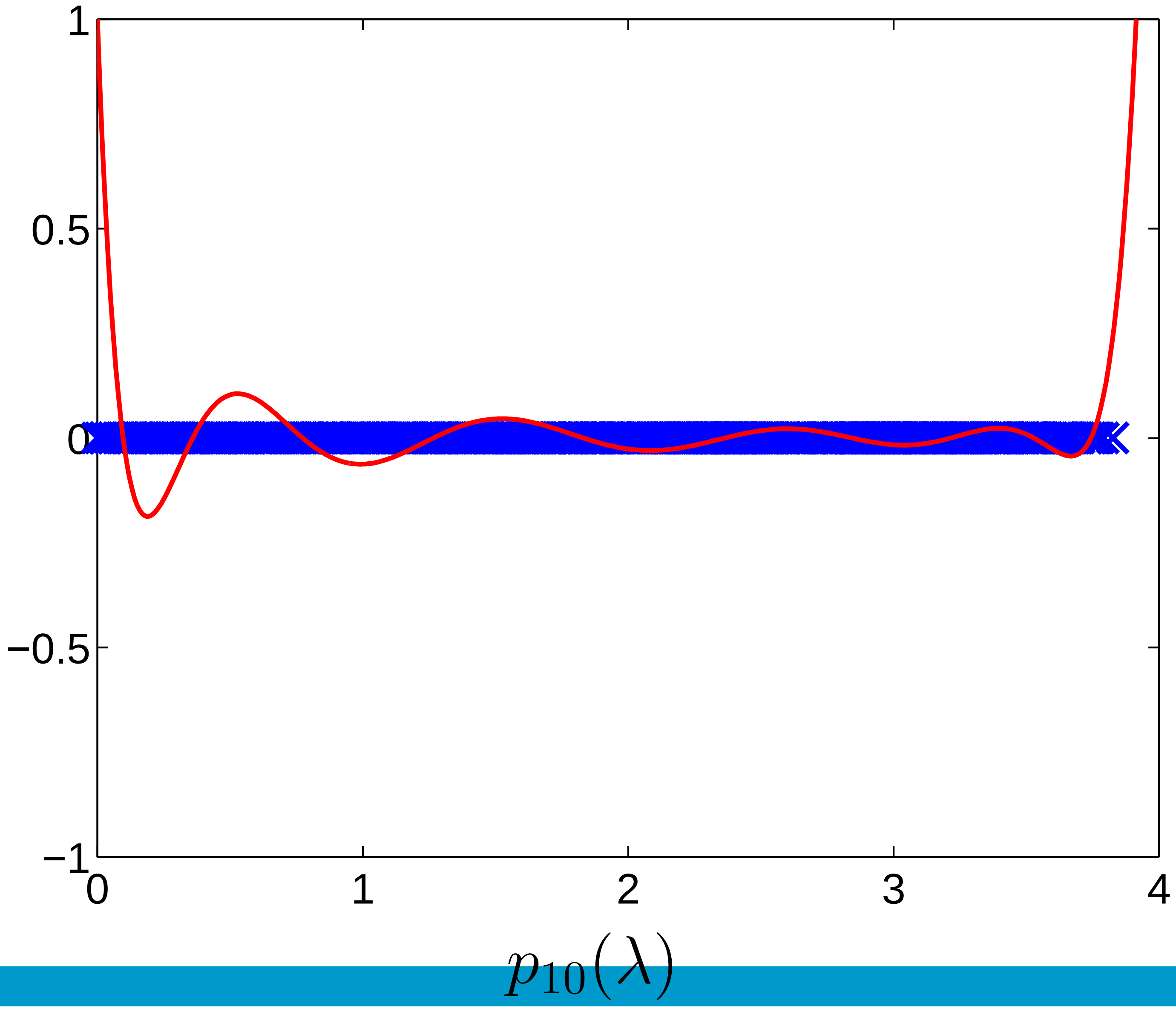
CG convergence: Dense Spectrum



CG convergence: Dense Spectrum



CG convergence: Dense Spectrum



Minimising the residuals

CG minimises the A-norm of the error. As we have seen before, another way to construct optimal approximations x_k is to minimise the residual, i.e. minimise

$$g(x_k) = \|A(x_k - x)\|_2^2 = r_k^T r_k$$

over all $x_k \in \{x_0 \cup K^k(A;b)\}$. Before we solve this problem we first look again at the Lanczos relation.

The Lanczos relation

By defining

$$\underline{T}_k = \begin{bmatrix} \alpha_1 & \beta_2 & & & 0 \\ \beta_2 & \alpha_2 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & 0 & \ddots & \ddots & \beta_k \\ & & & \beta_k & \alpha_k \\ & & & & \beta k + 1 \end{bmatrix}$$

the Lanczos relation can also be written as

$$AQ_k = Q_{k+1}\underline{T}_k$$

Minimal residuals

The problem is: find $x_k = x_0 + Q_k y_k$ such that $\|r_k\|$ is minimal.

$$r_k = b - Ax_k = r_0 - AQ_k y_k = \|r_0\| q_1 - AQ_k y_k$$

hence minimise

$$\begin{aligned}
\|r_k\| &= \| \|r_0\| q_1 - AQ_k y_k \| && (1)\\
&= \| \|r_0\| Q_{k+1} e_1 - Q_{k+1} \underline{T}_k y_k \| && (1)\\
&= \| \|r_0\| e_1 - \underline{T}_k y_k \|
\end{aligned}$$

Towards a practical algorithm

Solving the small overdetermined system $\underline{T}_k y_k = \|r_0\| e_1$ provides iterates

$$x_k = x_0 + Q_k y_k$$

that minimise the residual. The resulting algorithm is called MINRES.

MINRES can be cast in a CG like algorithm, see the books of van der Vorst and of Saad for details.

The resulting algorithm is called Conjugate Residual method.

Conjugate Residual Method

```
$r_0 = b - Ax_0;\quad p_0 = r_0$                                  initialization
FOR        $k = 0, 1, \cdots,$ DO
           $\alpha_k = \frac{r_k^T A r_k}{(Ap_k)^T A p_k}$
           $x_{k+1} = x_k + \alpha_k p_k$                         update iterate
           $r_{k+1} = r_k - \alpha_k A p_k$                       update residual
           $\beta_k = \frac{r_{k+1}^T A r_{k+1}}{r_k^T A r_k}$
           $p_{k+1} = r_{k+1} + \beta_k p_k$                      update direction vector
           $Ap_{k+1} = Ar_{k+1} + \beta_k A p_k$                  to avoid extra matvec
END FOR
```

Properties of CR

Like CG, CR has many favourable properties:

- The method uses limited memory: only four vectors need to be stored;
- The method is optimal: the residual is minimised;
- The method is finite: the n-st residual must be zero since it is optimal over the whole space;
- The method is robust if A is SPD, else $r_k^T A r_k$ may be zero for some nonzero r_k

CR is less popular than CG since minimising the A-norm of the error is often more natural. CG is also slightly cheaper.

CG for the normal equations

CG can always be applied to the normal equations

$$A^TAx = A^Tb \;\; (\text{or} \;\; AA^Ty = b \quad \text{with} \quad x = A^Ty)$$

since A^TA is an SPD-matrix (if A has full column rank).

Applying CG to the normal equations has two disadvantages:

- The work per iteration is twice as much as in CG;
- $K_2(A^TA) = K_2(A)^2$, which means that convergence is often very slow.

CGLS

CGLS is a numerical stable variant of CG for the normal equations.

The stability properties are improved by replacing inner products

$$p^T(A^TAp)$$

by inner products

$$(Ap)^TAp$$

LSQR

LSQR is derived by applying Lanczos to

$$\begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} r \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix} .$$

The resulting algorithm is equivalent to CGLS.

Concluding remarks

Today, we discussed Krylov methods for symmetric systems.

These methods combine an optimal error reduction with short recurrences, and hence limited memory requirements.

The next two days, we will discuss methods for solving nonsymmetric systems.