Numerical Linear Algebra for Computational Sciences and Information Engineering
CITHN2006

Basic Iterative Methods for Linear Systems

by Nicolas Venkovic

Chair of Computational Mathematics
School of Computation, Information and Technology (CIT)
Technical University of Munich, Germany

1. *Implement Jacobi, Gauss-Seidel and SOR iterations.*

[1]:
```
using LinearAlgebra, Plots, Printf, BenchmarkTools, SparseArrays
```

[2]:
```
function jacobi(A, b, x0, tol, max_iter)
  n, = size(b)
  x = copy(x0)
  xnew = similar(x0)
  err = zeros(Float64, max_iter + 1)
  b2norm = norm(b)
  err[1] = norm(b - A * x) / b2norm
  k = 1
  while (err[k] > tol && k < max_iter)
    k += 1
    xnew .= 0
    for i = 1:n
      for j = 1:n
        if j != i
          xnew[i] -= A[i, j] * x[j]
        end
      end
    end
    for i = 1:n
      xnew[i] = (xnew[i] + b[i]) / A[i, i]
    end
    x .= xnew
    err[k] = norm(A * x - b) / b2norm
  end
  return x, err[1:k]
```

```
end;
```

[3]:
```
function gauss_seidel(A, b, x0, tol, max_iter)
  n, = size(b)
  x = copy(x0)
  err = zeros(Float64, max_iter + 1)
  b2norm = norm(b)
  err[1] = norm(b - A * x) / b2norm
  k = 1
  while (err[k] > tol && k < max_iter)
    k += 1
    for i = 1:n
      x[i] = b[i]
      for j = 1:n
        if j != i
          x[i] -= A[i, j] * x[j]
        end
      end
      x[i] /= A[i, i]
    end
    err[k] = norm(A * x - b) / b2norm
  end
  return x, err[1:k]
end;
```

[4]:
```
function SOR(A, b, x0, tol, max_iter, omega = 1.5)
  n, = size(b)
  x = copy(x0)
  err = zeros(Float64, max_iter + 1)
  b2norm = norm(b)
  err[1] = norm(b - A * x0) / b2norm
  k = 1
  while (err[k] > tol && k < max_iter)
    k += 1
    for i = 1:n
      w = b[i]
      for j = 1:n
        w -= A[i, j] * x[j]
      end
      x[i] += omega * w / A[i, i]
    end
    err[k] = norm(A * x - b) / b2norm
  end
  return x, err[1:k]
end;
```

2. *Benchmark your implementations.*

[5]:
```
n = 5_000;
A = spdiagm(n, n, 0=>2.0*ones(n), 1=>0.5.*ones(n - 1), -1=>0.5.*ones(n - 1))
```

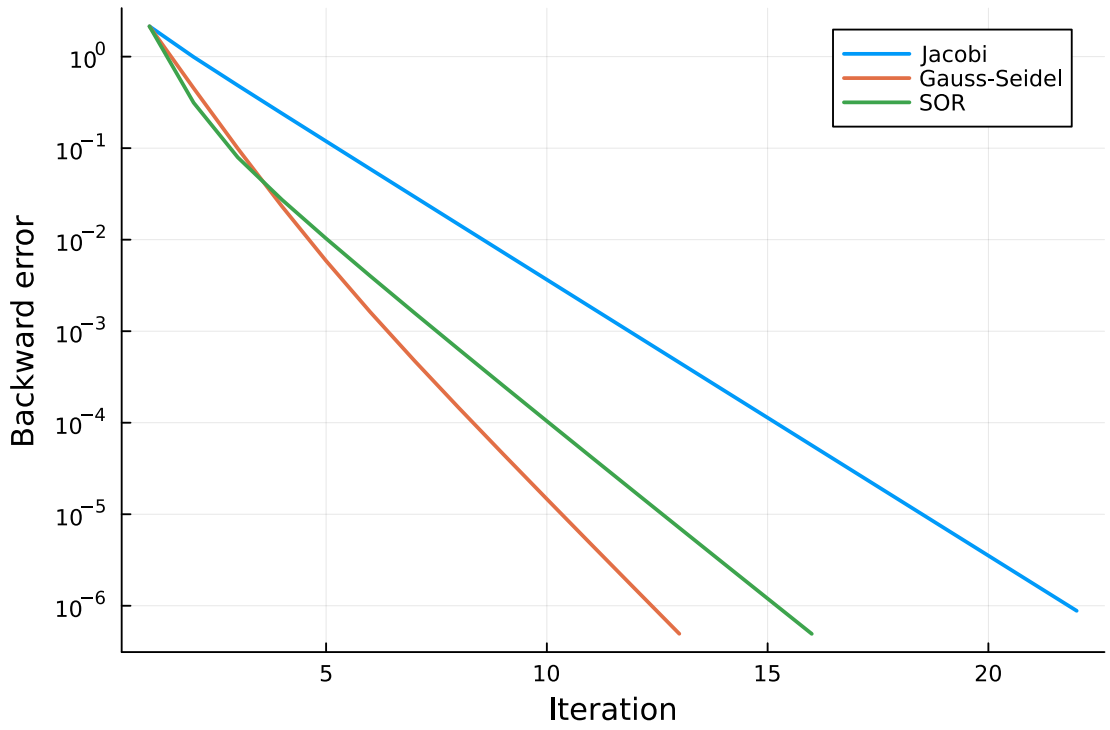
[5]:
```
5000×5000 SparseMatrixCSC{Float64, Int64} with 14998 stored entries:
```

[6]:
```
b = rand(n);
x0 = rand(n);
max_iter = 100;
tol = 1e-6;
```

[7]:
```
_, err_jacobi = jacobi(A, b, x0, tol, max_iter);
_, err_gs = gauss_seidel(A, b, x0, tol, max_iter);
_, err_sor = SOR(A, b, x0, tol, max_iter, .9);
```

[8]:
```
plot([i for i in 1:length(err_jacobi)], err_jacobi, yscale=:log10,␣
 ↪xlabel="Iteration", ylabel="Backward error", label="Jacobi", linewidth=2)
plot!([i for i in 1:length(err_gs)], err_gs, yscale=:log10,␣
 ↪label="Gauss-Seidel", linewidth=2)
plot!([i for i in 1:length(err_sor)], err_sor, yscale=:log10, label="SOR",␣
 ↪linewidth=2)
```

[8]:


Jacobi
Gauss-Seidel
SOR
Backward error
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
5
10
15
20
Iteration