Gauss-Seidel Iterative Method

Lecture 19 | Iterative methods for linear systems

Core idea	Improve an approximate solution one component at a time.
Key distinction	Use each newly computed value immediately in the same sweep.
Learning focus	Derivation, worked examples, convergence and Python implementation.

Learning outcomes

By the end of this lecture, students should be able to:

Derive	Obtain the sequential update equations from Ax = b.
Compute	Perform complete sweeps and construct numerical iteration tables.
Evaluate	Check convergence, residuals and differences from Jacobi.

Jacobi versus Gauss-Seidel

Both methods solve Ax = b by repeated approximation.

Jacobi	All components use values from the previous sweep.
Gauss-Seidel	Earlier components use new values; later components still use old values.
Implication	Gauss-Seidel is typically sequential and often converges faster, but not always.

Derivation of the update

Assume each diagonal entry a_{ii} is nonzero.

Row i $a_{i1}x_1 + \dots + a_{ii}x_i + \dots + a_{in}x_n = b_i$

Isolate x_i $x_i = (b_i - \Sigma\, a_{ij}x_j) / a_{ii}$, summed over $j \neq i$.

Sequential formula $x_i^{(k+1)} = [b_i - \Sigma(j<i)\, a_{ij}x_j^{(k+1)} - \Sigma(j>i)\, a_{ij}x_j^{(k)}] / a_{ii}$.

Matrix formulation

Split A = D + L + U, with actual off-diagonal entries in L and U.

Triangular system	$(D + L)x^{(k+1)} = b - Ux^{(k)}$.
Iteration	$x^{(k+1)} = (D + L)^{-1}(b - Ux^{(k)})$.
Iteration matrix	$B = -(D + L)^{-1}U$; $c = (D + L)^{-1}b$. Solve triangular systems rather than forming an inverse.

Algorithm and sequential updates

One full sweep updates components 1 through n.

Initialize	Choose $x^{(0)}$, tolerances and maximum sweeps; check nonzero diagonal.
Update	For i = 1,...,n, replace x_i using the latest available components.
Repeat or stop	After each full sweep, check step size and residual; report failure if the limit is reached.

Example 1 — two variables

Solve $4x + y = 9$ and $x + 3y = 7$, starting at $(0, 0)$.

Rearrange $x^{(k+1)} = (9 - y^{(k)})/4$; $y^{(k+1)} = (7 - x^{(k+1)})/3$.

First iteration $x^{(1)} = 9/4 = 2.25$; $y^{(1)} = (7 - 2.25)/3 = 1.583333$.

Second iteration $x^{(2)} = (9 - 1.583333)/4 = 1.854167$; $y^{(2)} = 1.715278$.

Example 1 — iteration table

Keep full precision internally; display rounded values.

k = 0, 1, 2	(0, 0) → (2.250000, 1.583333) → (1.854167, 1.715278)
k = 3, 4, 5	(1.821181, 1.726273) → (1.818432, 1.727189) → (1.818203, 1.727266)
Exact solution	x = 20/11 ≈ 1.818182; y = 19/11 ≈ 1.727273.

Example 2 — three variables

Start from $x^{(0)} = (0, 0, 0)$.

System	$10x - y + 2z = 6$; $-x + 11y - z = 25$; $2x - y + 10z = -11$.
Update equations	$x' = (6 + y - 2z)/10$; $y' = (25 + x' + z)/11$; $z' = (-11 - 2x' + y')/10$.
First iteration	1 | 0.600000 | 2.327273 | -0.987273 | 4.301818

Example 2 — second iteration

Use the newest x and y as soon as they become available.

x update $(6 + 2.327273 - 2(-0.987273))/10 = 1.030182$

y update $(25 + 1.030182 - 0.987273)/11 = 2.276628$

z update $(-11 - 2(1.030182) + 2.276628)/10 = -1.078374$

Example 2 — subsequent iterations

Columns: k | x | y | z | residual infinity norm.

k = 3, 4	3 | 1.043338 | 2.269542 | -1.081713 | 0.003340 4 | 1.043297 | 2.269235 | -1.081736 | 0.000262
k = 5, 6	5 | 1.043271 | 2.269230 | -1.081731 | 0.000014 6 | 1.043269 | 2.269231 | -1.081731 | 0.000000
Limit	$x \approx 1.043269$; $y \approx 2.269231$; $z \approx -1.081731$.

Convergence conditions

Strict diagonal dominance is sufficient, not necessary.

Strict dominance	For every row i: $\lvert a_{ii}\rvert > \Sigma \lvert a_{ij}\rvert$, $j \neq i$.
Example 2	10 > 3; 11 > 2; 10 > 3. Convergence follows from any starting vector.
General criterion	Convergence for every initial vector iff $\rho(B) < 1$. Symmetric positive definite A also guarantees convergence.

Error and stopping criteria

A small update alone does not guarantee an accurate solution.

Step size	$s_k = \|x^{(k+1)} - x^{(k)}\|_\infty$. Compare with a suitable tolerance.
Residual	$r = b - Ax$. Check $\|r\|_\infty$ after every full sweep.
Error bound	If $q = \|B\|_\infty < 1$, then $\|x^* - x^{(k+1)}\|_\infty \le q\, s_k/(1 - q)$.

Divergence and limitations

A nonsingular matrix need not yield a convergent iteration.

Counterexample	x + 2y = 3; 2x + y = 3. Start from (0,0).
Iterates	(0,0) → (3,−3) → (9,−15) → (33,−63). Exact solution: (1,1).
Diagnosis	B = [[0,−2],[0,4]] has spectral radius 4. Row ordering can affect convergence.

Computational cost and comparison

Compare work per sweep and the total number of sweeps.

Dense / sparse	$O(n^2)$ arithmetic per dense sweep; $O(\text{nnz}(A))$ for sparse storage.
Memory and ordering	In-place updates require one solution vector; order matters. Parallelization is less direct than Jacobi.
Example 1	Jacobi spectral radius = $1/\sqrt{12}$; Gauss-Seidel = $1/12$. This example converges faster with Gauss-Seidel.

Python implementation

NumPy example: A and b are floating-point arrays with nonzero diagonal.

Initialize

```
x = np.zeros(len(b))
for k in range(1000):
    old = x.copy()
```

Sequential sweep

```
    for i in range(len(b)):
        x[i] = (b[i]-A[i,:i]@x[:i]-A[i,i+1:]@old[i+1:])/A[i,i]
```

Stop inside loop

```
    if max(abs(x-old)) < 1e-8 and max(abs(b-A@x)) < 1e-8: break
else: raise RuntimeError("No convergence")
```

Practice questions

Attempt the exercises before viewing the solutions.

Q1 For 5x + y = 11 and x + 4y = 9, compute two Gauss-Seidel sweeps from (0,0).

Q2 Does A = [[4,1],[2,3]] satisfy strict diagonal dominance?

Q3 Explain why Gauss-Seidel can overwrite the solution vector in place.

Solutions and recap

Rearrange → update sequentially → check convergence.

Q1	(2.2, 1.7) → (1.86, 1.785). Exact solution: (35/19, 34/19).
Q2 / Q3	Yes: 4 > 1 and 3 > 2. In-place updates are intentional: later equations need the newest values.
Next lecture	NC_20 — Diagonal Dominance, Conditioning and Convergence.