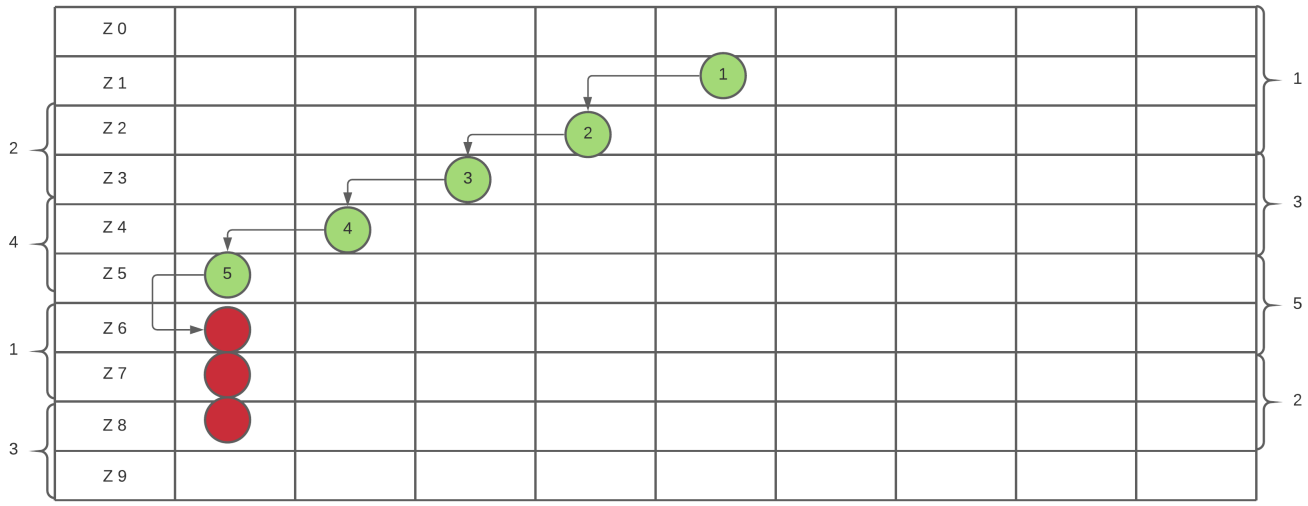
Gauss-Seidel



Da das Verfahren mit genau einer Matrix arbeitet, muss die Matrix aufgrund der Datenabhängikeiten des Verfahrens zeilenweise auf die Prozesse aufgeteilt werden, um eine Parallelisierung zu ermöglichen. Prozess 1 berechnet die Zeile 1 und erhält dafür die Werte der Zeilen 0, 1 und 2. Prozess 2 berechnet die Zeile 2 und erhält dafür die Wertze der Zeilen 2 und 3. Da die Berechnungen von den Ergebnissen der ersten Zeile abhängen, kann Prozess 2 erst mit den Berechnungen anfangen, wenn Prozess 1 das betreffende Element aus der ersten Zeile übermittelt hat.

Der Ablauf sieht folgendermaßen aus:
Prozess 1 startet mit der Berechnung des Elements (1, 1). Sobald das Ergebnis feststeht, übermittelt Prozess 1 dieses an Prozess 2 und fährt mit dem Element (1, 2) fort.
Prozess 2 wartet auf das Ergebnis vom Element (1, 1). Sobald dieses erhalten wurde, fängt Prozess 2 mit der Berechnung von (2, 1) an und übermittelt das Ergebnis an Prozess 3. Prozess 2 wartet nun auf Prozess 1 für das Ergebnis von (1, 2), um dann (2, 2) berechnen zu können.

Da die Matrix mehr Zeilen hat als es Prozesse gibt, muss jeder Prozess mehrere Zeilen berechnen. Nach Beendung der Zeile 1 startet Prozess 1 nun in einer späteren Zeile (in unserem Beispiel: Zeile 6).
Für eine Leistungsoptimierung könnten wir sicherstellen, dass die letzten beiden Zeilen vom letzten Prozess bearbeitet werden. Dadurch könnte mit der nächsten Iteration begonnen werden, bevor die erste abgeschlossen ist.

Für das Zusammentragen der Ergebnisse können wir die Werte nach Rang der Prozesse entgegennehmen. Da jeder nachfolgende Prozess die Werte des Vorgängers erhalten hat, würden beim möglichen Überschreiben keine Informationen verloren gehen.

Abbruchbedingungen:

Iterationen:

Jeder Prozess erhält einen eigenen Iteration-Counter. Diesen erhöht er für die jeweils letzte zu berechnende Zeile um 1. In unserem Beispiel würde also Prozess 1 den Counter erst erhöhen, nachdem er Zeile 6 fertig berechnet hat.

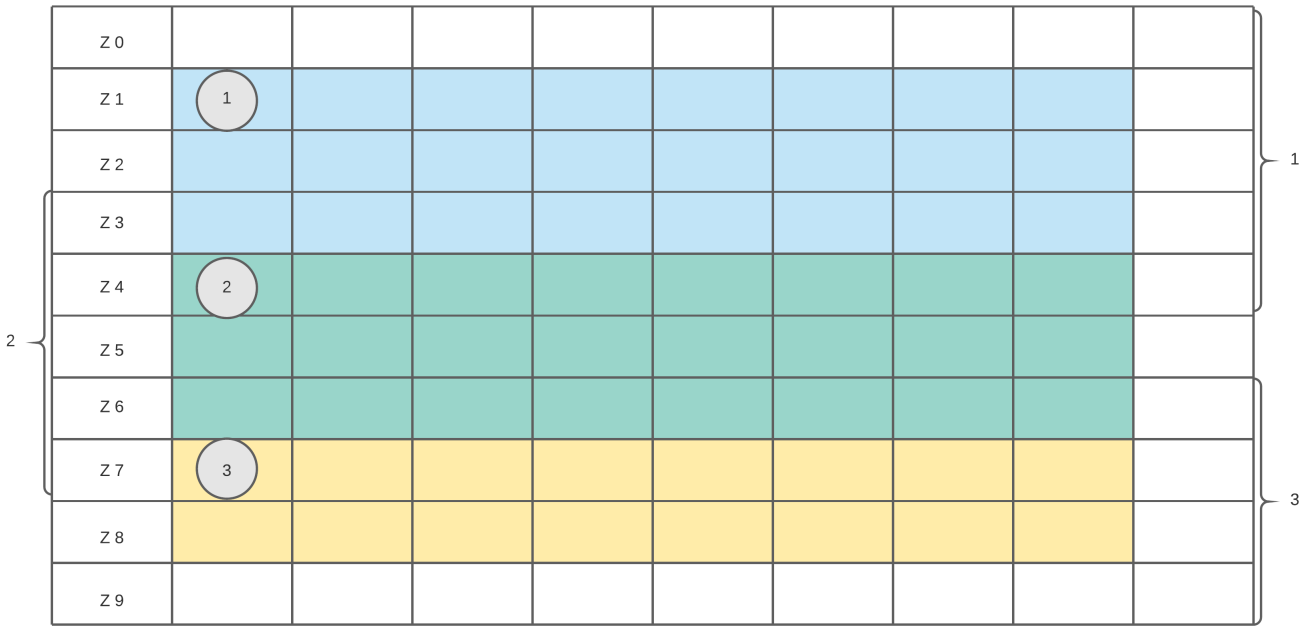
Erreicht der Counter den gewünschten Wert, übermittelt der jeweilige Prozess seine Daten an einen ausgewählten Prozess zum Zusammenführen der Ergebnisse.

Genauigkeit:

Ein Abbruch nach Genauigkeit setzt vorraus, dass die gesamte Matrix mindestens bis auf eine gewisse Genauigkeit hin berechnet wurde. Dies beudetet, dass wir ledigleich die letzte Zeile auf die Genauigkeit überprüfen müssen, da die vorherigen Zeilen mindestens die gleiche Genauigkeit haben. Wird die gewünschte Genauigkeit in der letzten Zeile erreicht, übermittelt der betreffende Prozess das Abbruchsignal an alle anderen Prozesse, welche dann mit den Berechnungen aufhören und mit dem Zusammentragen der Ergebnisse beginnen.

Die Überprüfung, ob die Genauigkeit erreicht wurde, erfolgt vor dem Start der Berechnung einer Zeile. Auch denkbar wäre die Überprüfung vor jeder Iteration, wodurch das Abbruchsignal allerdings recht spät wahrgenommen wird und viele überflüssige Berechnungen erfolgen, oder eine Überprüfung vor jeder Element-Berechnung, welches allerdings einen hohen Aufwand für das konstante Abfragen bedeuten würde. Mit der Überprüfung vor jeder Zeile erhoffen wir, einen Mittelweg zwischen diesen beiden Nachteilen zu finden.

Jacobi



Das Jacobi-Verfahren speichert die berechneten Ergebnisse in einer weiteren Matrix. Dies ermöglicht eine Aufteilung der Matrix in Blöcke, wobei jeder Prozess einen anderen Block bearbeitet. In unserem Beispiel würde Prozess 1 die Zeilen 1 bis 3 bearbeiten und dafür die Zeilen 0 bis 4 erhalten, Prozess 2 würde die Zeilen 1 bis 6 berechnen und dafür die Zeilen 3 bis 7 erhalten, und so weiter.

Für das Abspeichern der berechneten Werte würde jeder Prozess eine weitere Matrix bereitstellen. Nach Beendung einer Iteration würden die Werte der Ergebnis-Matrix in die betreffenden Stellen der ausgehenden Matrix eingetragen werden. Hierbei würden die Prozesse außerdem ihre Rand-Zeilen an die benachbarten Prozesse senden.

In unserem Beispiel würde Prozess 1 die neu berechnete Zeile 3 an Prozess 2 senden, Prozess 2 würde die neue Zeile 4 an Prozess 1 und die neue Zeile 6 an Prozess 3 senden, Prozess 3 würde die neue Zeile 7 an Prozess 2 senden.

Abbruchbedingungen:

Iterationen:

Wie auch beim Gauss-Verfahren würde jeder Prozess einen eigenen Iterations-Counter verwalten, den er nach dem Versenden der Rand-Zeilen erhöht. Erreicht dieser Counter den gewünschten Wert, würden die betreffenden Prozesse mit dem Zusammentragen der Ergebnisse anfangen. Da wir den Counter erst nach dem Versenden/Erhalten der benachbarten Zeilen erhöhen, können wir die Ergebnisse einfach nach Rang der Prozesse zusammentragen, da jeder nachfolgende Prozess die gleichen Werte in der betreffenden Zeile hat und somit auch bei Überschreibungen keine Informationen verloren gehen.

Genauigkeit:

Da durch das Versenden/Empfangen der Rand-Zeilen mit MPI_Send/MPI_Recv eine Synchronisierung erfolgt, arbeitet jeder Prozess mit der gleichen Genauigkeit. Damit kann auch jeder Prozess seine eigene Genauigkeit bestimmen und diese nach dem Versenden/Empfangen seiner Rand-Zeilen überprüfen. Ist der gewünschte Wert erreicht, fangen alle Prozesse mit dem Zusammentragen der Ergebnisse an.