Numerisches Lösen – Iterative Nullstellenberechnung

1. **Vorgehensmodell**

   Als Vorgehensmodell habe ich Kanban gewählt, da es eine klare visuelle Darstellung des Projektfortschritts ermöglicht und gleichzeitig hohe Flexibilität bietet. Das es besonders einfach zu implementieren ist kann es besonders in diesem kleinen Umfang schnell eingebaut werden.

2. **Programmier-Paradigma (Strukturiert, Objektorientiert)**

   Als Programmier-Paradigma habe ich mich für das Objektorientierte Programmierung entschieden um den Code möglichst übersichtlich, lesbar und modularisierbar zu gestallten.

3. **Zeitmanagement**

Erste Berechnungen in Excel, um Bisektion zu verstehen (Aufgabe 1 & 2)	3 Stunden
PAP zu dem Bisektionsproblem erstellen (Aufgabe 3)	1 Stunde
Planungsarbeiten (Aufgabe 4)	2 Stunden
Programmierung sowie Tests in Python (Aufgabe 5)	2.5 Stunden
Alternative Lösungsmöglichkeit (Aufgabe 6)	3 Stunden
Diagramme mittels matplotlib (Aufgabe 7)	1.5 Stunden
Polynom Berechnung (Aufgabe 8)	1.5 Stunden
Elektrische Leitung Berechnung (Aufgabe 9)	5 Stunden
Summe	**19.5 Stunden**

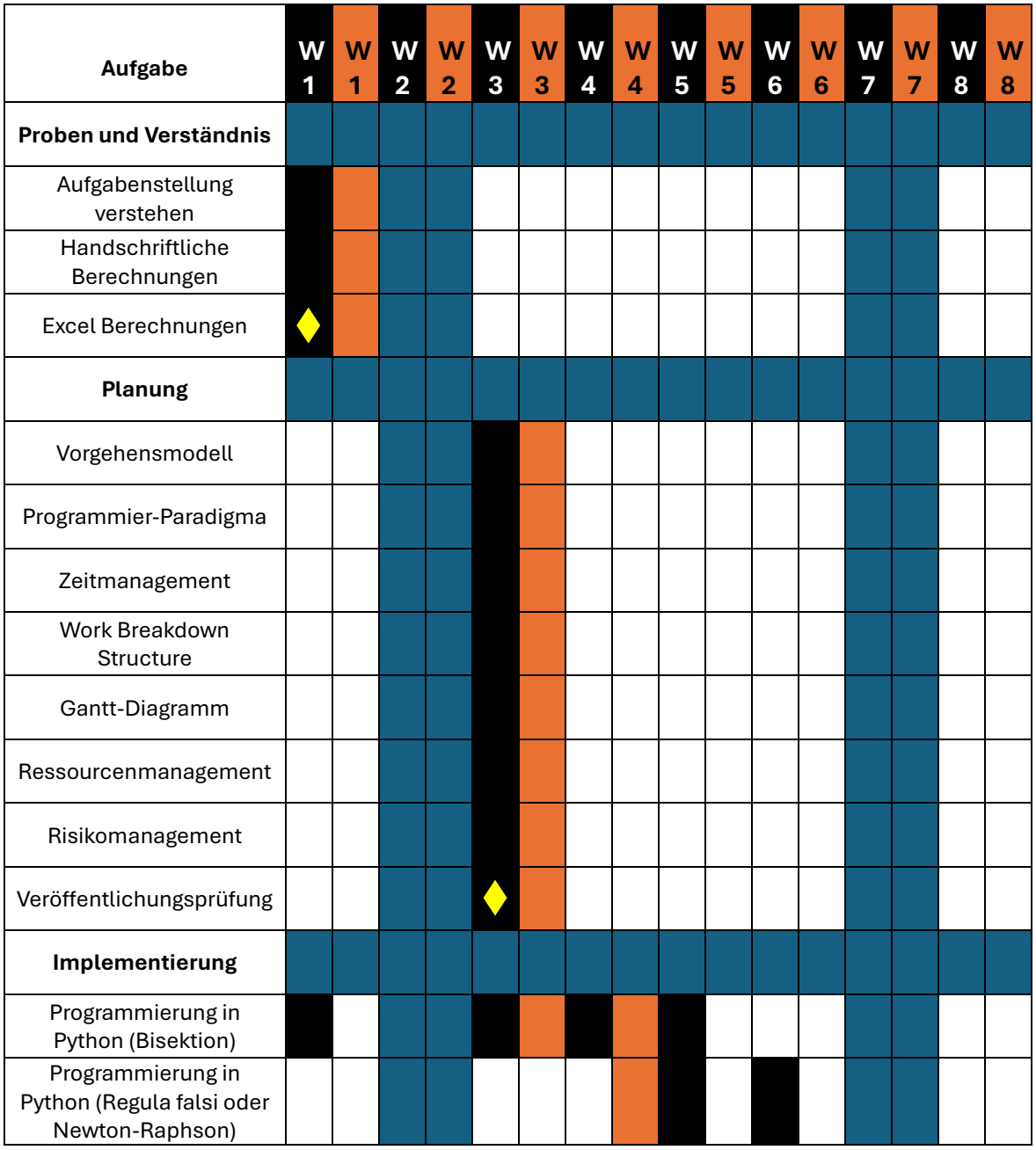
4. **Work Breakdown Structure (WBS)**
   a. Proben und Verständnis
      i. Aufgabenstellung verstehen
      ii. Handschriftliche Berechnungen
      iii. Excel Berechnungen
   b. Planung
      i. Vorgehensmodell
      ii. Programmier-Paradigma
      iii. Zeitmanagement
      iv. Work Breakdown Structure
      v. Gantt-Diagramm
      vi. Ressourcenmanagement
      vii. Risikomanagement
      viii. Veröffentlichungsprüfung
   c. Implementierung
      i. Programmierung in Python (Bisektion)
      ii. Programmierung in Python (Regula falsi oder Newton-Raphson)
      iii. Matplotlib Programmierung
      iv. Polynom Programmierung in Python

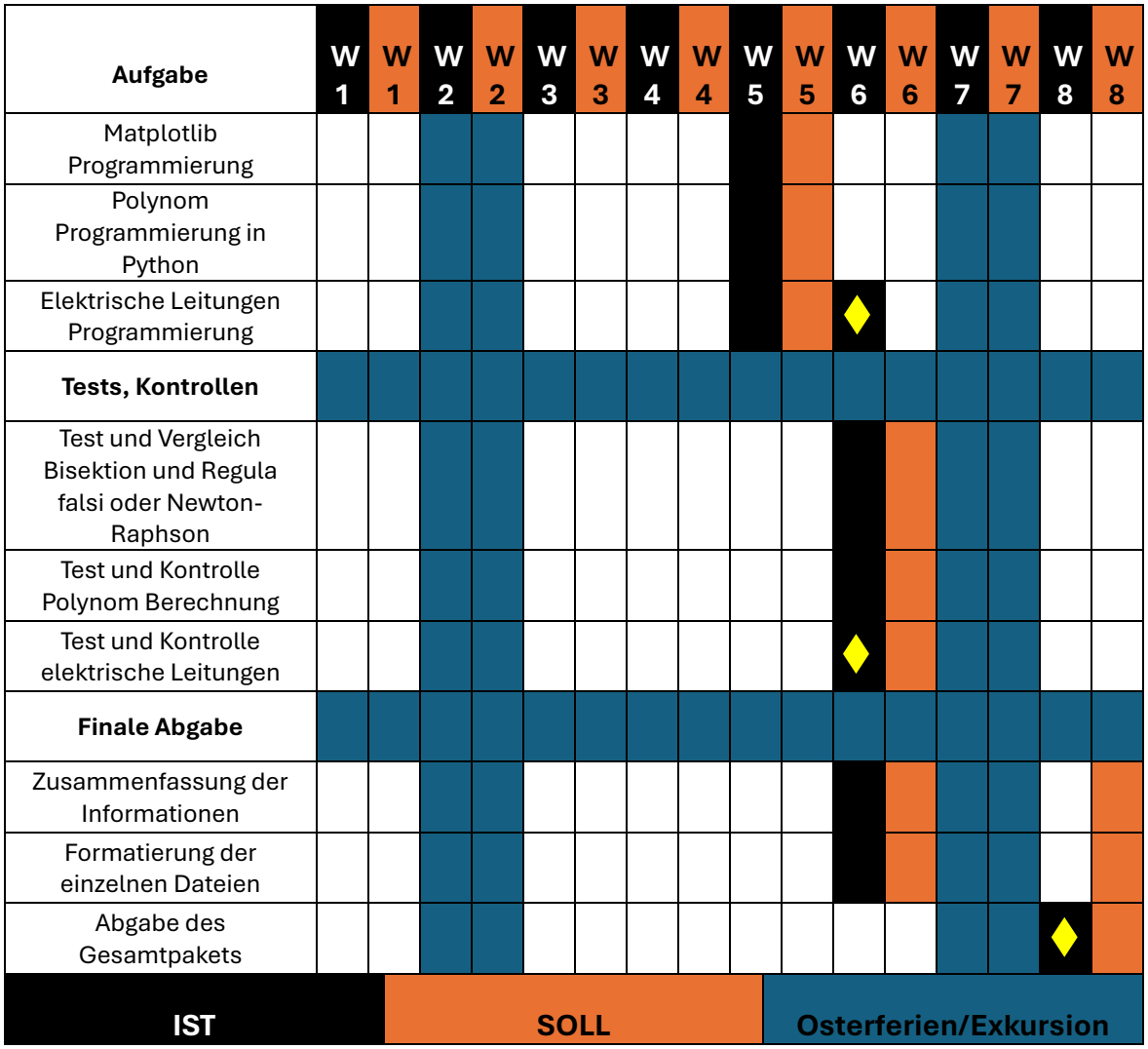
v. Elektrische Leitungen Programmierung in Python

d. Tests, Kontrollen
   i. Test und Vergleich Bisektion und Regula falsi oder Newton-Raphson
   ii. Test und Kontrolle Polynom Berechnung
   iii. Test und Kontrolle elektrische Leitungen Berechnung

e. Finale Abgabe
   i. Zusammenfassung der Informationen
   ii. Formatierung der einzelnen Dateien
   iii. Abgabe des gesamt Packets

5. Gantt-Diagramm





♦ **... Meilensteine**

1. Abschluss der Proben
2. Abschluss der Planung
3. Abschluss der Implementierung
4. Abschluss der Tests und Kontrollen
5. Finale Abgabe

6. **Ressourcenmanagement**
    a. Personal:
        i. 1 Entwickler (Ich)
    b. Hardware
        i. Laptop
        ii. Ladegerät
        iii. Maus
        iv. Papier
        v. Stift
        vi. Taschenrechner

c. Software
    i. Python
    ii. Matplotlib
    iii. Word
    iv. Excel
    v. Google

7. Risikomanagement

a. Interne Risiken
    i. Unterschätzer Aufwand -> Pufferzeit einplanen
    ii. Speicherverlust -> Cloud-Speicher / GitHub-Repo
    iii. Hardwareprobleme -> Ersatzhardware und Cloud-Speicher
    iv. Fehlerhafter Code -> Stack Overflow / Google / Professoren bei Fragen
b. Externe Risiken
    i. Blackout in Österreich -> Zeitverlust
    ii. Software Ausfälle -> Wechsel auf andere Software
    iii. Umweltkatastrophen -> Ausfall von Arbeit im Gesamten

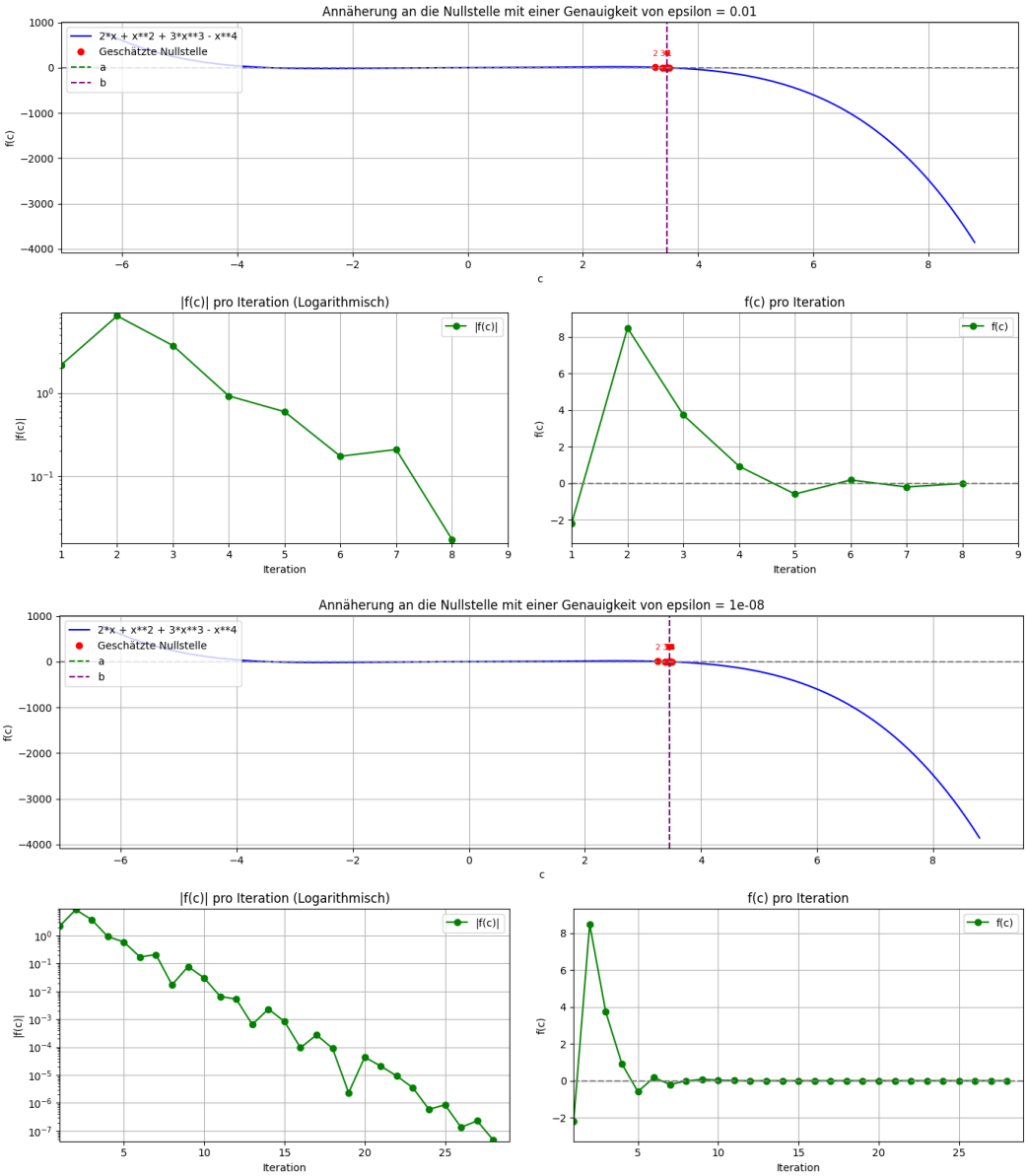
8. Veröffentlichungsprüfung

a. Matplotlib: Open-Source-Bibliothek -> Mit Python unter PSF-Lizenz
b. Visual Studio Code (VSC): Open-Source-Editor unter MIT-Lizenz
    i. Verbreitung: Matplotlib und Python können frei verwendet, verändert und weitervergeben werden, solange die Lizenzbedingungen der Software eingehalten werden. Da VSC nicht mitgeliefert wird fällt die Lizenz hierbei weg.

9. Programmierung

- **Aufgabe 8:**

  Da die zu findende Nullstelle bei x = 3,4567 liegt kann zu der Berechnung einfach als Intervall **[3, 4]** angenommen werden. Zu beachten ist hierbei, dass ein Startintervall benötigt wird welcher in der ersten Berechnung a und b als Werte mit unterschiedlichen Vorzeichnen berechnet (P4(3) = 15, P4(4) = -40). Ansonsten kann der Intervall-Switch nicht erfolgreich durchgeführt werden. 0 und 4 würden daher nicht funktionieren.
  Die Berechnung mit einer Genauigkeit von $\varepsilon = 10^{-2}$ benötigt mit dem Intervall [3, 4] **8 Iterationen**. Bei einer Genauigkeit von $\varepsilon = 10^{-8}$ sind es **28 Iterationen**. → Es wird bei den Iterationen auch a_0 und b_0 also die initiale Berechnung als Iterationsschritt gezählt. Ansonsten wären es 7 und 27 Iterationen.



- **Aufgabe 9:**

  Im ersten Schritt muss eine passende Funktion für die Nullstellenberechnung erstellt werden. Dafür wird in der Gleichung der Kettenlinie y(x) 50 eingesetzt also $y(50) = \alpha * \cosh(50 / \alpha) - \alpha + y_0$ dies ermöglicht es die Gleichung der Kettenlinie mit der Randbedingung gleichzustellen. → $y(50) = \alpha * \cosh(50 / \alpha) - \alpha + y_0 = y_0 + 10$ von dieser Funktion kann dann die passende Funktion für die Nullstellenfindung abgeleitet werden. → $f(x) = x * \cosh(50 / x) - x - 10$ Diese Funktion kann dann mithilfe des Bisektionsverfahrens und einem Intervall von [100, 200] berechnet werden. Dieses Intervall kommt aus etwas Trial-and-Error. → 100m wären zu viel (in der Formel eingesetzt wären das ca. 12,8m statt den vorgeschriebenen 10m) dies bedeutet, dass der gesuchte Radius größer als 100m sein muss. Die

nächstgrößere Zahl, welche zu einem sinnvollen Ergebnis führt, war in meinen Berechnungen 200 (nach der Berechnung mit [100, 200] ist mir aufgefallen, dass 150 genauso funktioniert hätte). Eine Berechnung durch die Bisektion mit dem Startintervall **[100, 200]** und der Genauigkeit von $\varepsilon = 10^{-50}$ bringt ein Ergebnis von ca. **126,63m** als Krümmungsradius α. Zur Probe wurde dieses Ergebnis dann als α in die oben gleichgestellte Formel eingesetzt. Als Ergebnis kam ca. 10m raus also die korrekte Biegung der Leitung. Zu guter Letzt konnte ich nun die Länge der Leitung mit der Gleichung l = 2 * α * sinh(100 / 2 * α) berechnen. Dies führte zu einem Ergebnis von ca. **102,62m** als Länge der Leitung.

10. Einsatz von KI

a. KI wurde in diesem Projekt nur für Folgende Zwecke eingesetzt:
   - Kommentieren des Codes (in simplen kurzen Stichworten – komplexere Kommentare wurden somit von mir selbst geschrieben)
   - Docstrings der Klassen und Funktionen – hierfür habe ich ein Schema erstellt, welches die KI auf jegliche von mir ausgewählten Funktionen und Klassen angewendet hat.
   - Für die Plots – da die Aufgabenstellung nicht eindeutig war habe ich KI Plots generieren lassen, bis diese denen im Unterricht von Herr Professor Schobesberger vorgestellten entsprachen. Für weiter Informationen hierzu siehe „TESTING“ Ordner in der Abgabe.

b. Verwendete KIs:
   - ChatGPT
   - Mistral
   - DeepSeek
   - GitHub Copilot
   - Gemini