高校数学とJulia言語 Day 3

関数の最大・最小を求めよう！

城北中学校・高等学校　中学3年・高校1年
夏期講習会III 2025/8/24～2025/8/28
担当：清水団

📅 5日間の学習予定

- **Day 1**：Google Colabの紹介・基本計算 ✅
- **Day 2**：関数のグラフの描画 ✅
- **Day 3**：最適化（最大・最小） ← 今日はここ！
- **Day 4**：データの分析
- **Day 5**：確率・シミュレーション

> 今日のゴール：グラフを見て最大・最小を見つけ、数値的に求められるようになろう！

🎯 最大・最小を求める意味

数学だけでなく、実生活でも重要！

実生活での応用例

- 📈 **経済学**：利益を最大化、費用を最小化
- 🏗️ **工学**：材料を最小限で最大の強度を得る
- ⚽ **スポーツ**：最高記録を出すための軌道
- ⏰ **日常生活**：時間や労力を最小にして最大の効果を得る

> グラフを描いて視覚的に理解し、その後数値的に正確な値を求める方法を学びます！

🔧 今日使うツール

昨日と同じツールを使います

```julia
# パッケージの読み込み
using Plots

# フォント設定（日本語ラベルのため）
gr(fontfamily="ipam")
```

準備完了！さっそく最適化問題に挑戦しましょう

📊 基本的な最適化：2次関数

問題1：$f(x) = -x^2 + 4x + 1$ **の最大値を求める**

Step 1: まずグラフで全体像を把握

```
# 関数を定義
f(x) = -x^2 + 4x + 1

# グラフを描画
plot(f, lw=3, color=:blue, label="f(x) = -x² + 4x + 1")
```

グラフから最大値の位置を予想してみましょう！

🔍 数値的に最大値を探す

Step 2: 予想した範囲を詳しく調べる

```
# x = 2 付近を詳しく調べる
X = 1.5:0.01:2.5    # x = 1.5 から 2.5 まで 0.01 刻み
Y = f.(X)           # 各x値での関数値を計算

# 最大値とその位置を見つける
y_max = maximum(Y)
x_max = X[argmax(Y)]

println("x = $x_max のとき, 最大値 y = $y_max")
```

> 数値計算で正確な値が求まります！

📈 グラフに最大値をプロット

Step 3: 結果を視覚化

```
# グラフに最大値の点を追加
plot(f, lw=3, label="f(x) = -x² + 4x + 1", color=:blue)
scatter!([x_max], [y_max], ms=8, color=:red,
         label="最大値 ($x_max, $y_max)")
```

理論的解法との比較

$$\text{平方完成：} f(x) = -(x-2)^2 + 5$$
$$\therefore \max f(x) = f(2) = 5$$

数値解と理論解が一致！

📉 制約付き最適化：3次関数

問題2：$g(x) = x^3 - 6x^2 + 9x + 2$ の区間 $[-1, 3]$ での最大・最小

3次関数では極大値・極小値が複数存在することがあります

```
# 関数を定義
g(x) = x^3 - 6x^2 + 9x + 2

# 広い範囲でグラフを描いて全体像を把握
plot(g, -2:0.1:4, lw=3, color=:green)

# 制約区間 [-1, 3] を表示
vline!([-1, 3], color=:red, lw=2, alpha=0.7)
```

🎯 制約区間での数値探索

指定された区間内だけで探索

```
# 区間 [-1, 3] での数値的探索
X = -1:0.01:3
Y = g.(X)

# 最大値と最小値を見つける
y_max = maximum(Y)
y_min = minimum(Y)
x_max = X[argmax(Y)]
x_min = X[argmin(Y)]

println("最大値: ", y_max, " (x = ", x_max, ")")
println("最小値: ", y_min, " (x = ", x_min, ")")
```

制約条件がある場合は端点も要チェック！

三角関数の最適化

問題3：$h(x) = 2\sin(x) + \cos(2x)$ **の区間** $[0, 2\pi]$ **での最大・最小**

周期的な関数では複雑なパターンが現れます

```
# 関数を定義
h(x) = 2sin(x) + cos(2x)

# グラフを描いて周期的パターンを観察
plot(h, 0:0.01:2π, lw=3, color=:purple)

# x軸の目盛りをπ単位で表示
xticks!([0, π/2, π, 3π/2, 2π],
        ["0", "π/2", "π", "3π/2", "2π"])
```

📊 三角関数の数値解析

より細かく刻んで精密に探索

```
# 区間 [0, 2π] での数値的探索
X = 0:0.001:2π  # より細かく刻んで探索
Y = h.(X)

y_max = maximum(Y)
y_min = minimum(Y)
x_max = X[argmax(Y)]
x_min = X[argmin(Y)]

# π単位での表示
println("最大値の位置: x ≈ ", round(x_max/π, digits=3), "π")
println("最小値の位置: x ≈ ", round(x_min/π, digits=3), "π")
```

複雑な関数でも数値的手法で解けます！

⚖️ 手法の比較

数値的手法（今日の方法）

メリット：

✅ 複雑な関数でも適用可能
✅ グラフで視覚的に理解
✅ プログラムで自動化
✅ 制約条件を簡単に扱える

デメリット：

❌ 近似解しか得られない
❌ 計算時間がかかる場合
❌ 局所最適解を見落とす可能性

理論的手法（微分使用）

メリット：

✅ 厳密解が得られる
✅ 計算が高速
✅ 数学的に美しい

デメリット：

❌ 微分可能な関数に限定
❌ 複雑な関数では計算困難
❌ 制約条件の扱いが複雑

🔄 数値的最適化の手順

標準的なアプローチ

1. **グラフの描画**：関数の全体像を把握
2. **視覚的予想**：最大・最小の位置を大まかに特定
3. **詳細探索**：予想した範囲を細かく刻んで数値計算
4. **結果の検証**：理論値や他の方法と比較

重要なポイント

- **定義域の確認**：制約条件を常に意識
- **端点のチェック**：境界での値も忘れずに確認
- **グラフの活用**：視覚的理解が問題解決の鍵

📝 実習：最適化を体験しよう

実際にGoogle Colabで以下を試してみましょう

1. **2次関数の最適化**：グラフ→予想→数値探索
2. **制約付き問題**：区間を限定した最適化
3. **複雑な関数**：三角関数や合成関数の最適化
4. **結果の可視化**：最適点をグラフにプロット

> 手を動かして体験することが一番の学習です！

📚 本日の演習問題

問題1: 2次関数の最小値

$f(x) = 2x^2 - 8x + 11$ の最小値とそのときのxの値を求める

問題2: 3次関数の制約付き最適化

$g(x) = x^3 - 3x^2 + 2$ の区間$[-1, 3]$での最大値・最小値を求める

問題3: 応用問題

$h(x) = \sqrt{x^2 + 1} + \sqrt{x^2 - 2x + 5}$ の最小値を求める

> 各問題で：グラフ描画 → 数値探索 → 結果の可視化 → 考察

🎯 演習問題を解いてみよう！

Google Colabを開いて、実際にコードを書いてみましょう

取り組み方

1. **関数を定義**してグラフを描く
2. **視覚的に予想**する
3. **数値的に探索**する
4. **結果を文章で出力**する
5. **グラフに最適点をプロット**する
6. **理論的解法と比較**する（可能であれば）

> エラーを恐れずに、いろいろ試してみましょう！

🌟 今日のまとめ

学んだ数値的最適化の手順

1. **グラフ描画**：関数の全体像を把握
2. **視覚的予想**：最大・最小の位置を特定
3. **詳細探索**：数値計算で正確な値を求める
4. **結果検証**：理論値との比較

扱った関数の種類

- **2次関数**：頂点での最大・最小
- **3次関数**：極値と制約条件
- **三角関数**：周期的な最大・最小

重要なスキル

✅ グラフを描いて問題を理解する能力
✅ 数値的手法で複雑な問題を解く技術
✅ 制約条件を適切に扱う方法

🔮 次回予告：Day 4

データの分析

- 実際のデータを使った統計分析
- 平均・分散・相関などの統計量の計算
- データの可視化とパターンの発見
- Juliaでのデータ処理技術

> 今まで学んだグラフ描画技術を使って、データの特徴を視覚的に分析します！

宿題

今日の演習問題を完成させて、Google Classroomに提出してください。

💡 発展的な内容（時間がある人へ）

より高度な最適化問題

```
# 多変数関数の最適化（等高線プロット）
f(x, y) = x^2 + y^2 - 2x - 4y + 5
x = -1:0.1:3
y = -1:0.1:5
contour(x, y, f, levels=20)

# 条件付き最適化
# 制約: x^2 + y^2 ≤ 1 の下で f(x,y) を最小化
```

より複雑な問題にも挑戦できます！

❓ 質問タイム

何か分からないことはありませんか？

- 最適化の手法について
- グラフの読み取り方
- 数値計算の精度
- 演習問題について
- その他、何でも！

> 最適化は実用性の高い分野です。疑問があれば積極的に質問しましょう！

お疲れさまでした！