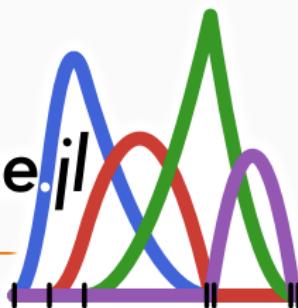


# BasicBSpline.jl で始める B-spline

---



堀川由人 (@Hyrodium)

2023 年 7 月 11 日

## スライド公開先



<https://www.docswell.com/s/hyrodium/5Q89MJ-B-spline>

# 目次

1. 自己紹介
2. B-spline 最速入門
3. B-spline もう少し入門
4. B-spline の応用例
5. まとめ

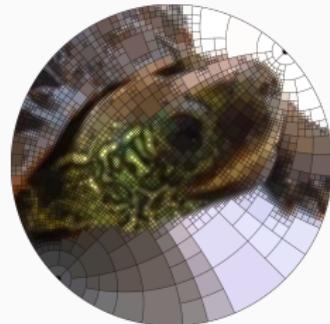
# 目次

1. 自己紹介
2. B-spline 最速入門
3. B-spline もう少し入門
4. B-spline の応用例
5. まとめ

# 経歴・所属

## 名前

- 堀川由人
- @Hyrodium



## 経歴

- 大阪府立工業高等専門学校
- 大阪大学(学部・修士)
- DMG 森精機

## 現所属

- Rist
- 大阪大学大学院基礎工学研究科  
(招聘研究員)



# Julia と私

Zenn.dev

- Julia の行列・ベクトルを完全に理解すっぞ !!
- FastGaussQuadrature.jl で数値積分しましょう
- Quaternions.jl をメンテナンスしてる話
- Julia 言語における中置演算子の扱い



GitHub

- JuliaGeometry/Rotations.jl
- JuliaApproximation/FastGaussQuadrature.jl
- JuliaArrays/StaticArrays.jl
- JuliaGeometry/Quaternions.jl
- JuliaMath/IntervalSets.jl
- hyrodium/ImageClipboard.jl
- hyrodium/**BasicBSpline.jl**



# B-spline と私

## これまでに執筆した記事など

- B-spline 入門（線形代数がすこし分かる人向け）  
B-spline のモチベーションを伝えるための動画
- NURBS 多様体による形状表現  
証明を詳しく書いた PDF 資料
- BasicBSpline.jl のドキュメント  
パッケージの使用方法と簡単な数学的解説
- BasicBSpline.jl を作ったので宣伝です！  
BasicBSpline.jl を作ったアナウンス on Zenn.dev
- 新しくなった BasicBSpline.jl(v0.8.3) の紹介  
バージョンが上がった件について解説
- なめらかな曲線を Julia で SVG 出力する  
BasicBSpline.jl の応用例の紹介記事
- BasicBSpline.jl - Basic mathematical operations for B-spline  
Discourse でのパッケージ告知
- Plotting smooth graphs with Julia  
「なめらかな曲線を Julia で SVG 出力する」の英訳版

# 目次

---

1. 自己紹介

2. B-spline 最速入門

B-spline とは？

関数近似

B-spline 基底関数

3. B-spline もう少し入門

4. B-spline の応用例

5. まとめ

# B-spline とは？

## 区分多項式を使って関数近似するための道具

- 多項式

閉区間上の任意の連続関数は多項式で近似可能  
(Weierstrass の多項式近似定理)

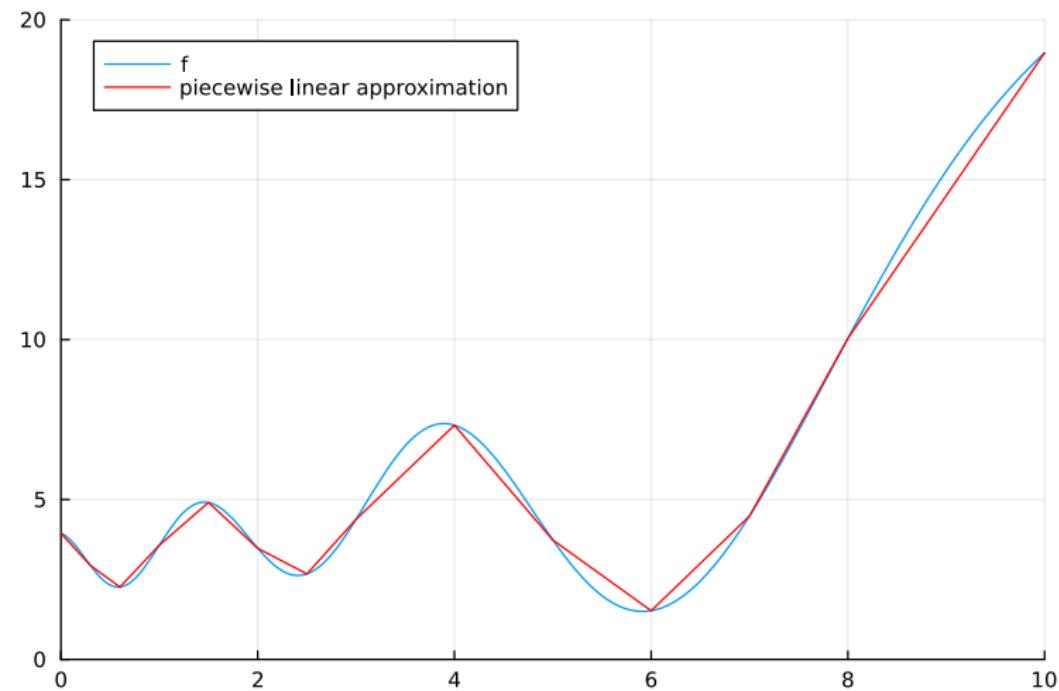
- 区分関数

定義域を適当な区間に分割し、各区間で関数を近似

B-spline で実現できること：

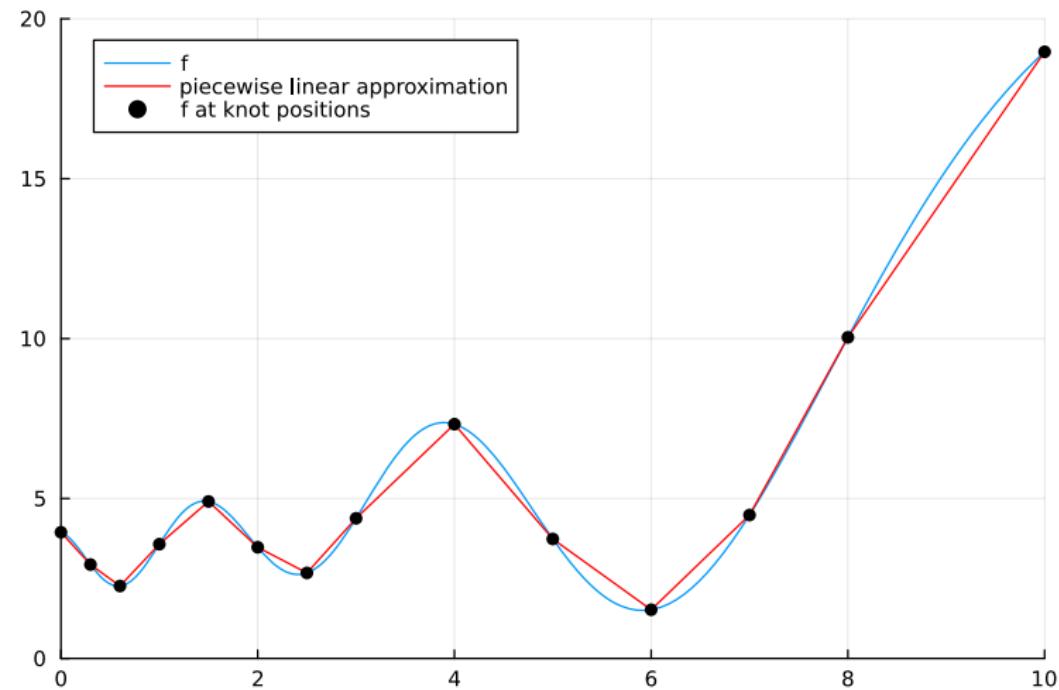
- 高すぎない多項式次数
- 多すぎない区間分割数
- 区分点で滑らかに繋げる

## 関数近似 ① (折れ線)



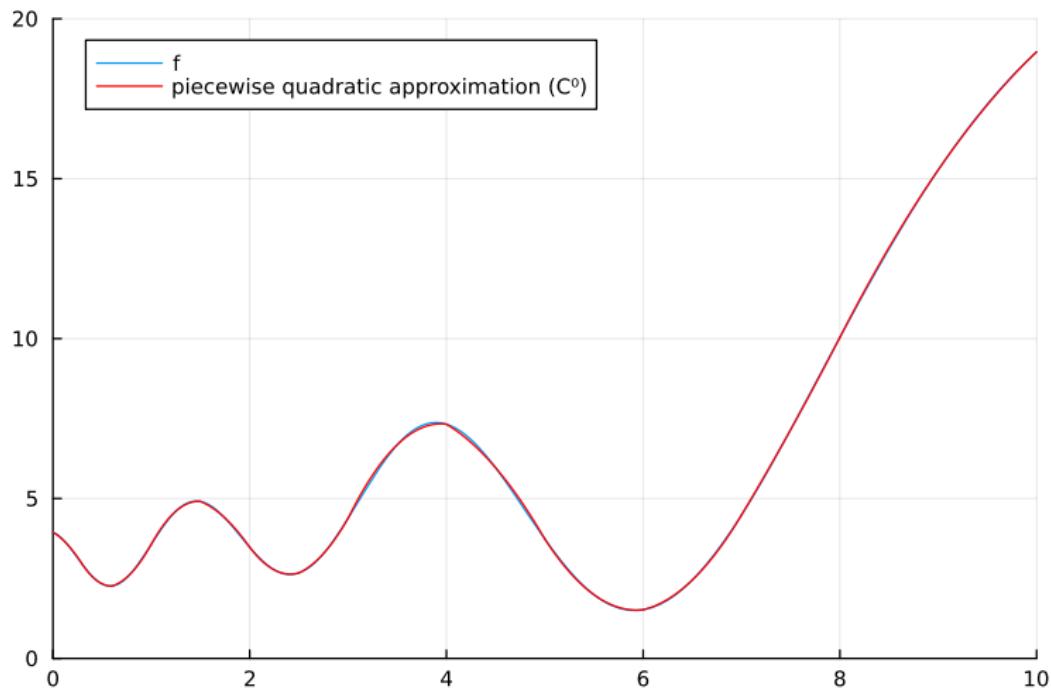
適当な点を折れ線で繋げば近似になる

## 関数近似 ① (折れ線)



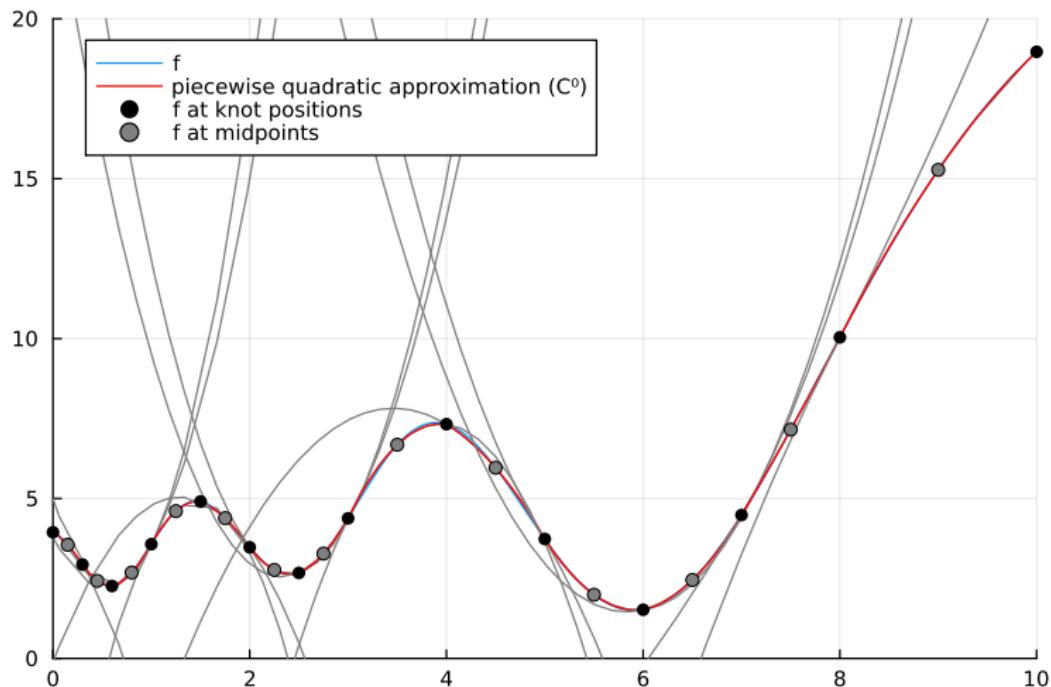
適当な点を折れ線で繋げば近似になる  
多項式の繋ぎ目(区分点)を「ノット(knot)」と呼ぶ

## 関数近似 ② (放物線)



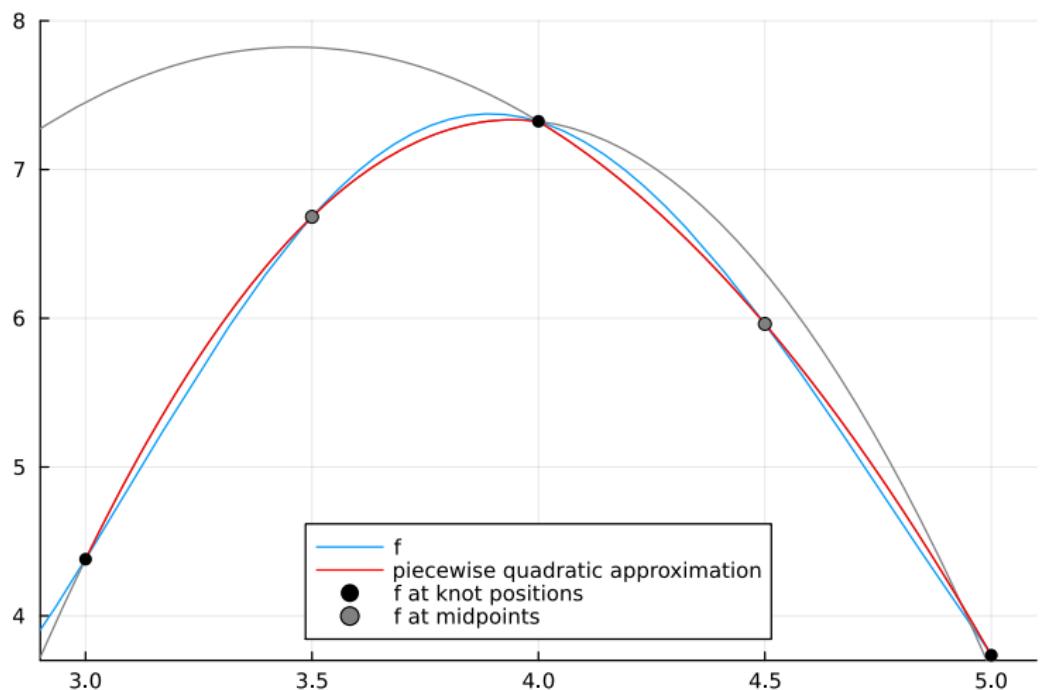
各区間で 3 点取って放物線 (2 次多項式) を描けば近似になる

## 関数近似 ② (放物線)



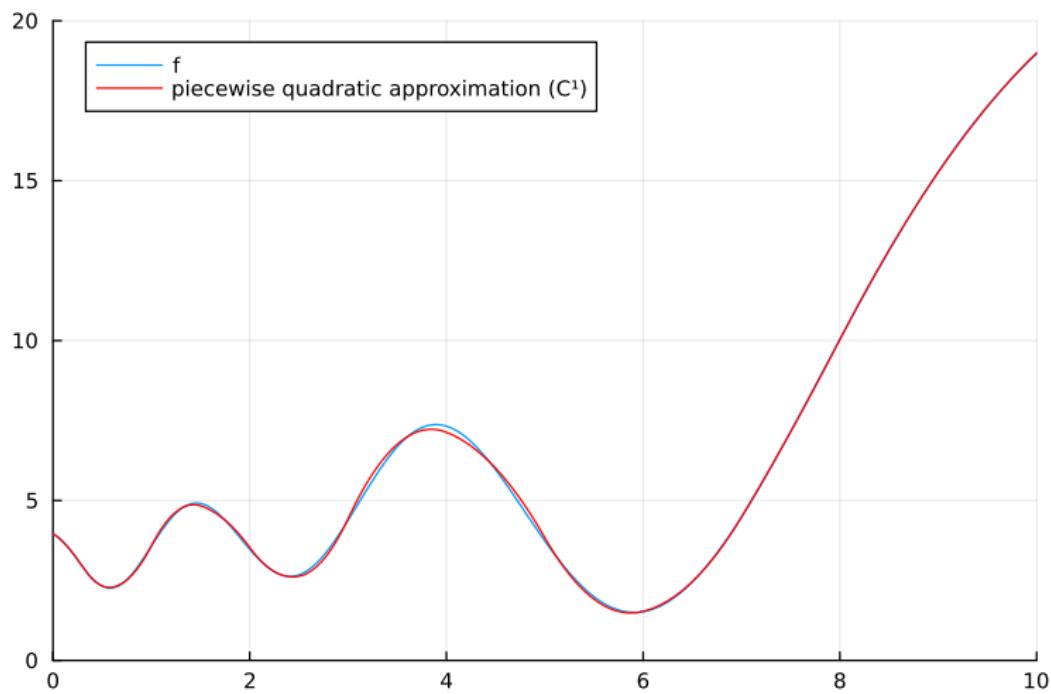
各区間で 3 点取って放物線 (2 次多項式) を描けば近似になる

## 関数近似 ② (放物線)



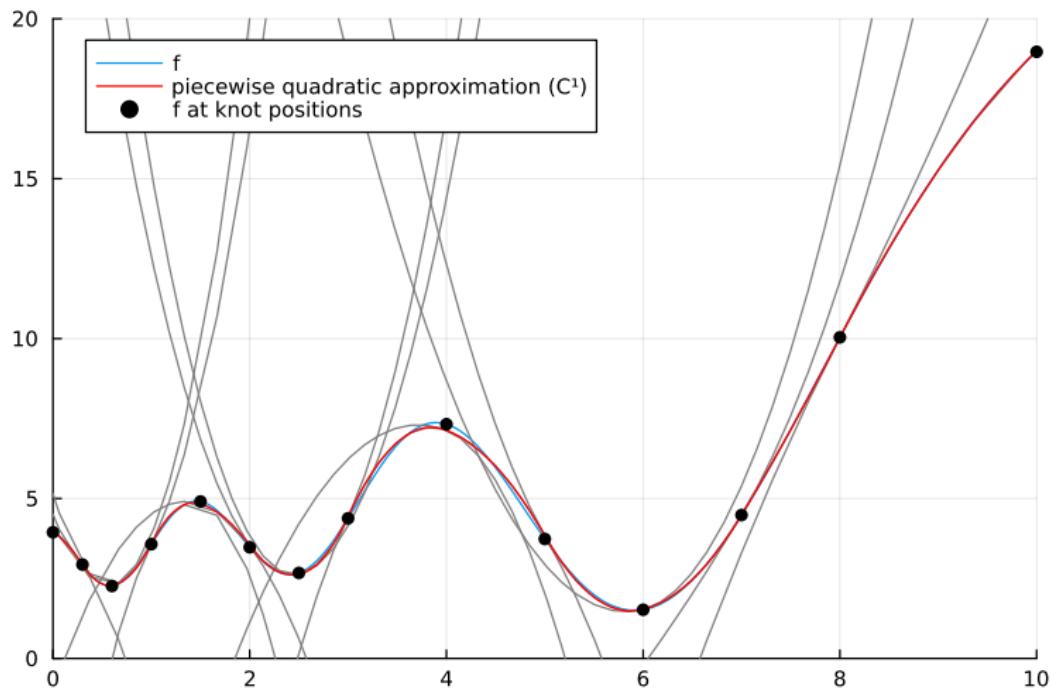
各区間で 3 点取って放物線 (2 次多項式) を描けば近似になる  
区分点で  $C^0$  級になる

### 関数近似 ③ (放物線、ただし $C^1$ 級)



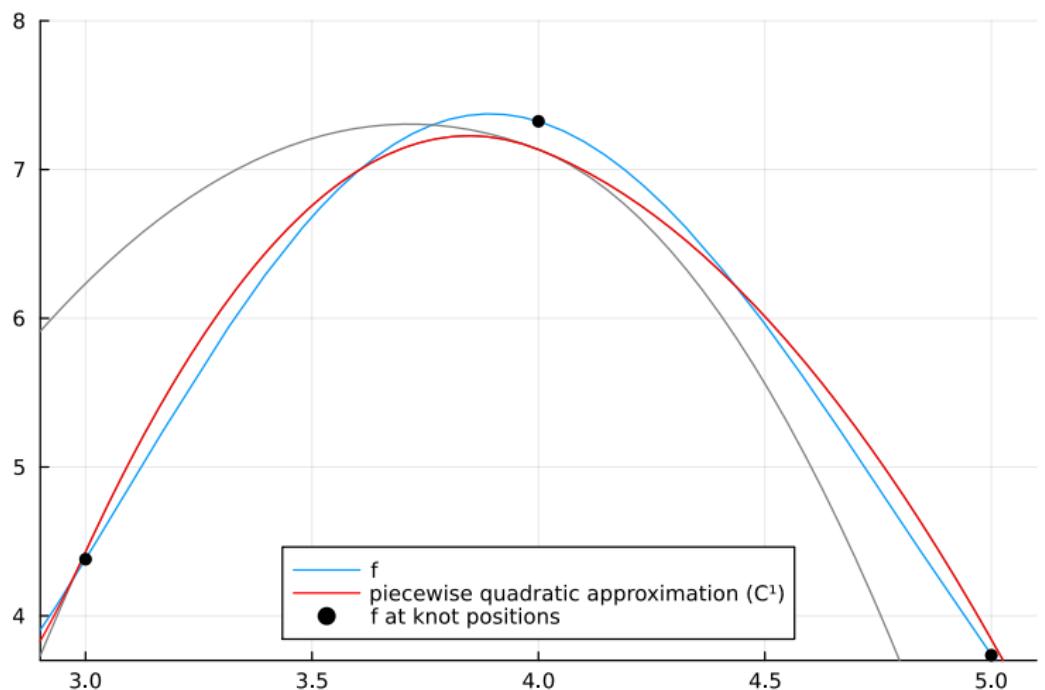
放物線で近似し、区分点で  $C^1$  級も要求

### 関数近似 ③ (放物線、ただし $C^1$ 級)



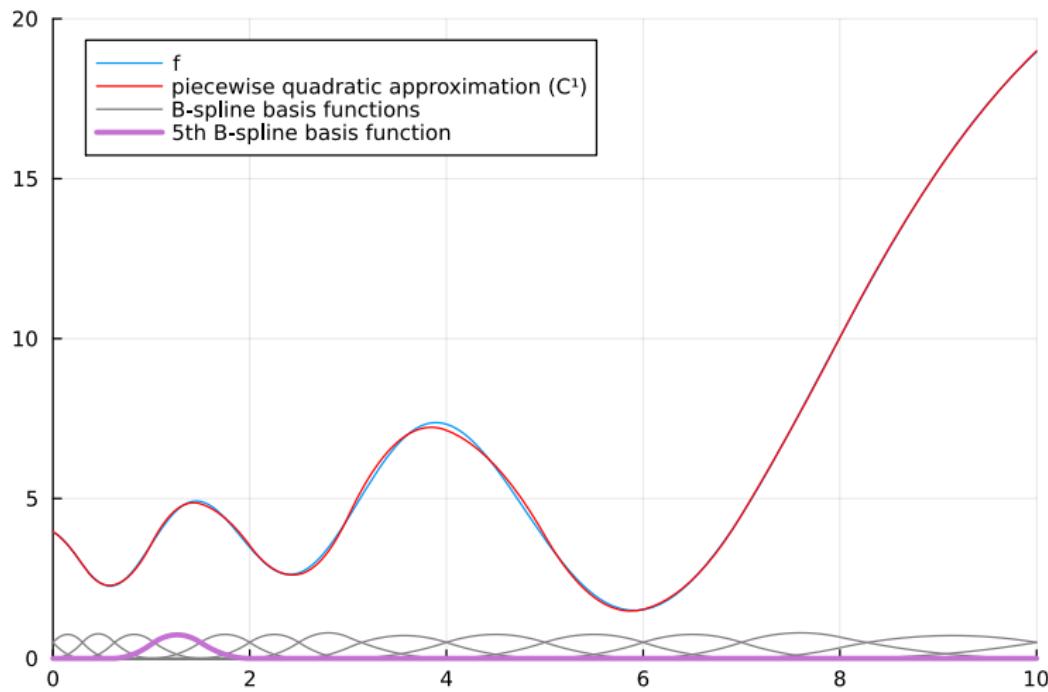
放物線で近似し、区分点で  $C^1$  級も要求

### 関数近似 ③ (放物線、ただし $C^1$ 級)



放物線で近似し、区分点で  $C^1$  級も要求  
→ どうすれば実現できる…？

## B-spline 基底関数 ① (近似の方針)



「滑らかさを保証した区分多項式」をたくさん用意して、線型結合すれば OK !  
これが B-spline 基底関数 !

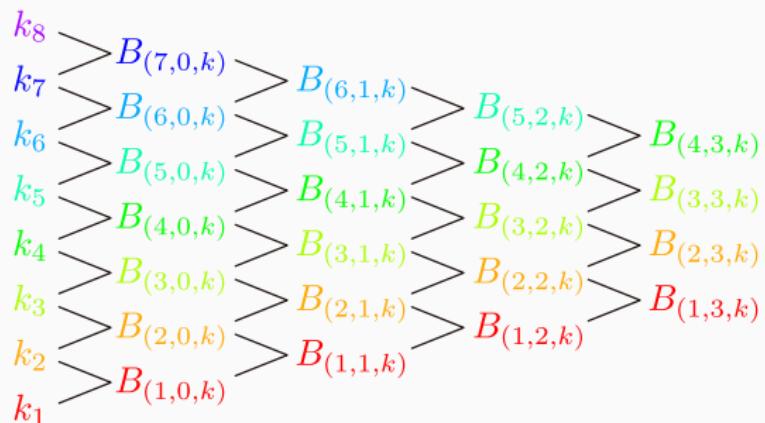
## B-spline 基底関数 ② (定義)

### 定義 (B-spline 基底関数)

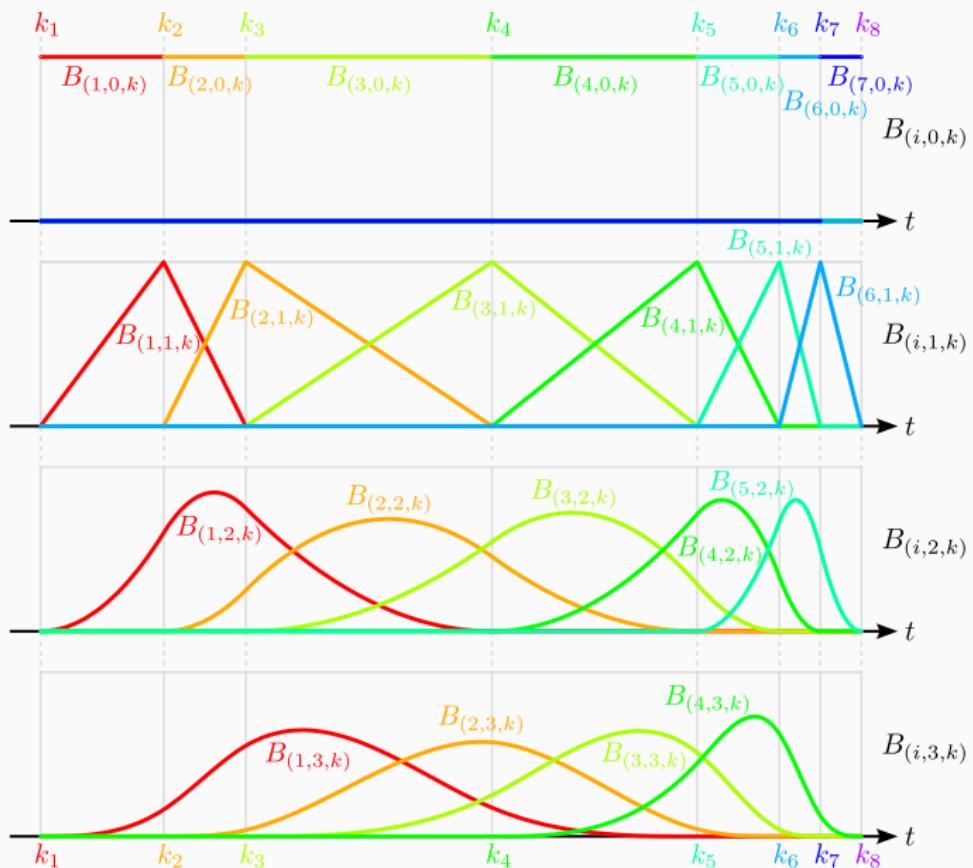
ノット列  $k = (k_1, \dots, k_l)$  に対する  $p$  次 B-spline 基底関数  $B_{(i,p,k)}$  は以下で定義される

$$B_{(i,p,k)}(t) = \frac{t - k_i}{k_{i+p} - k_i} B_{(i,p-1,k)}(t) + \frac{k_{i+p+1} - t}{k_{i+p+1} - k_{i+1}} B_{(i+1,p-1,k)}(t)$$

$$B_{(i,0,k)}(t) = \begin{cases} 1 & (k_i \leq t < k_{i+1}) \\ 0 & (\text{otherwise}) \end{cases}$$



## B-spline 基底関数 ② (定義)



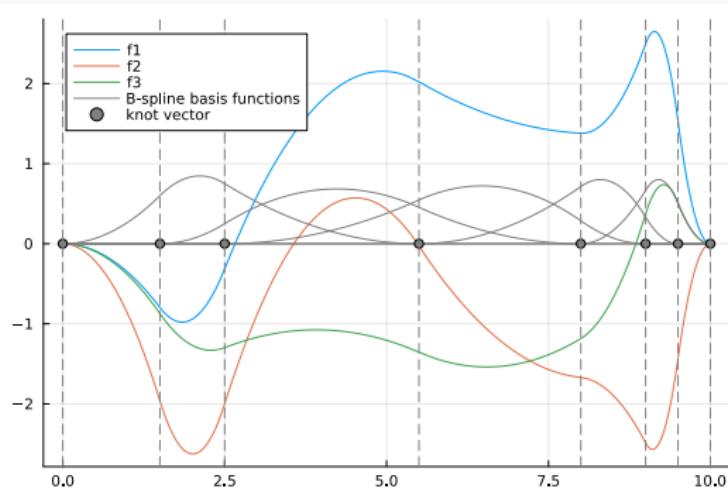
## B-spline 基底関数 ③ (B-spline 基底関数の張る空間)

### 定理 (B-spline 基底関数の張る空間)

狭義単調増加なノット列  $k = (k_1, \dots, k_l)$  で構成される B-spline 基底関数について以下が成立

$$\text{span}_i \{ B_{(i,p,k)} \} = \left\{ f \in C^{p-1}(\mathbb{R}) \mid \begin{array}{l} \forall i, f|_{(k_i, k_{i+1})} \text{ は } p \text{ 次多項式} \\ \text{supp}(f) \subseteq [k_1, k_l] \end{array} \right\}$$

この線形空間を  $\mathcal{P}[p, k]$  で表す



## B-spline 基底関数 ④ (疑問点いろいろ)

- 区分多項式空間  $\mathcal{P}[p, k]$  の次元は何で決まる？
- 2つの線形空間  $\mathcal{P}[p^1, k^1], \mathcal{P}[p^2, k^2]$  の包含関係は何で決まる？
- 特定のノット  $k_i$  での滑らかさの制約を変更できる？
- B-spline 基底関数  $B_{(i,p,k)}$  はどんな良い性質がある？
- B-spline 基底関数  $B_{(i,p,k)}$  の導関数は？
-

## B-spline 基底関数 ④ (疑問点いろいろ)

- 区分多項式空間  $\mathcal{P}[p, k]$  の次元は何で決まる？
- 2つの線形空間  $\mathcal{P}[p^1, k^1], \mathcal{P}[p^2, k^2]$  の包含関係は何で決まる？
- 特定のノット  $k_i$  での滑らかさの制約を変更できる？
- B-spline 基底関数  $B_{(i,p,k)}$  はどんな良い性質がある？
- B-spline 基底関数  $B_{(i,p,k)}$  の導関数は？
- **BasicBSpline.jl** で何ができるのか？

## B-spline 基底関数 ④ (疑問点いろいろ)

- 区分多項式空間  $\mathcal{P}[p, k]$  の次元は何で決まる？
- 2つの線形空間  $\mathcal{P}[p^1, k^1], \mathcal{P}[p^2, k^2]$  の包含関係は何で決まる？
- 特定のノット  $k_i$  での滑らかさの制約を変更できる？
- B-spline 基底関数  $B_{(i,p,k)}$  はどんな良い性質がある？
- B-spline 基底関数  $B_{(i,p,k)}$  の導関数は？
- BasicBSpline.jl** で何ができるのか？

```
using BasicBSpline, Plots
P = BSplineSpace{2}(KnotVector([0,1,2,3,4,5,6]))
n = dim(P)
P ⊂ BSplineSpace{2}(KnotVector(0:8))
plot(P)
plot!(BSplineSpace{2}(KnotVector([0,1,2,3,3,4,5,6])))
plot!(t -> sum(bsplinebasis(P,i,t) for i in 1:n))
plot!(BSplineDerivativeSpace{1}(P))
```

↑ 完全理解しなくて OK。雰囲気だけ

# 目次

---

1. 自己紹介
2. B-spline 最速入門
3. B-spline もう少し入門
  - B-spline 基底関数の性質
  - B-spline 多様体
  - リファインメント
4. B-spline の応用例
5. まとめ

# B-spline 基底関数の性質 ①

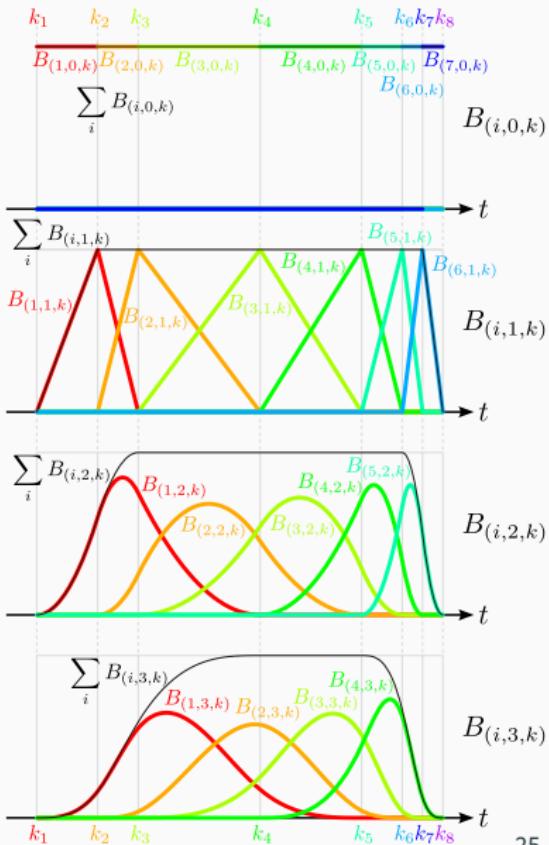
- ・台

$$\text{supp}(B_{(i,p,k)}) = [k_i, k_{i+p+1}]$$

- ・1 の分割

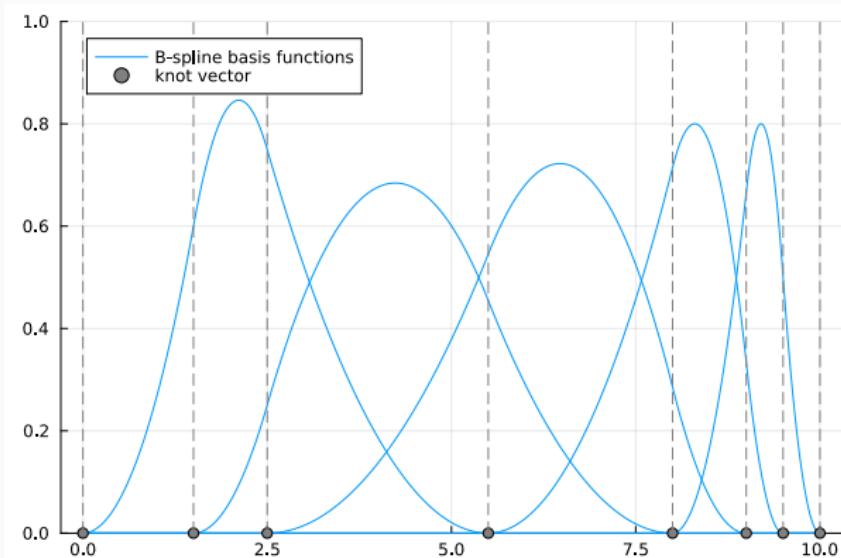
$$\sum_i B_{(i,p,k)}(t) = 1 \quad (t \in [k_{1+p}, k_{l-p}])$$

```
k = KnotVector(1:8)
P = BSplineSpace{2}(k)
n = dim(P)
plot(P)
scatter!(
    collect(k),
    zeros(length(k))
)
bsplinesupport(P, 2)
plot!(t->sum(
    bsplinebasis(P, i, t)
    for i in 1:n)
)
```



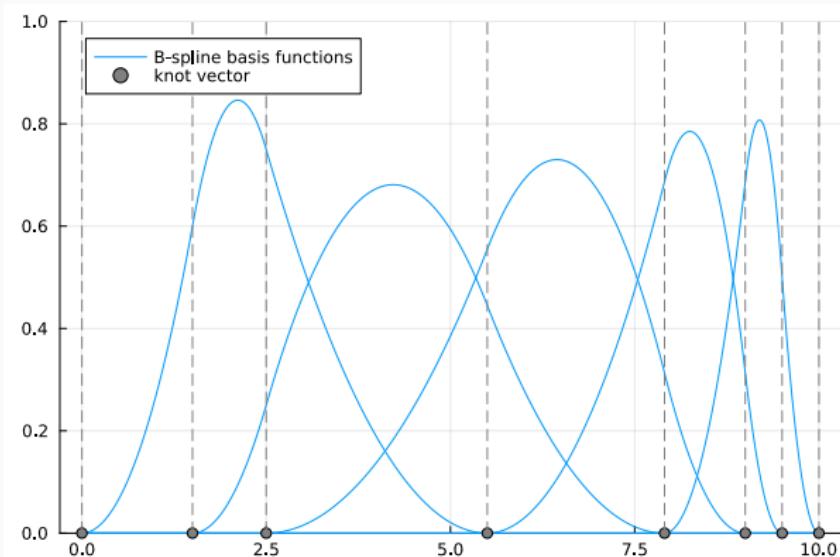
## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



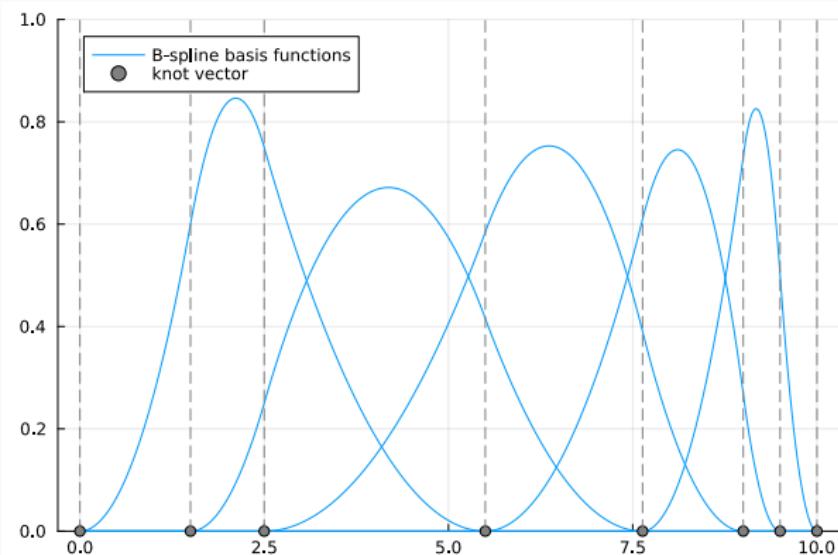
## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



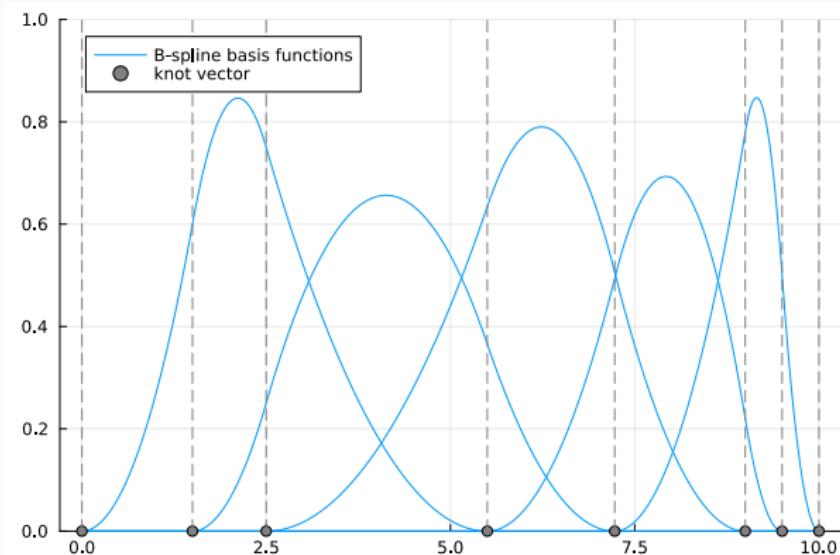
## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



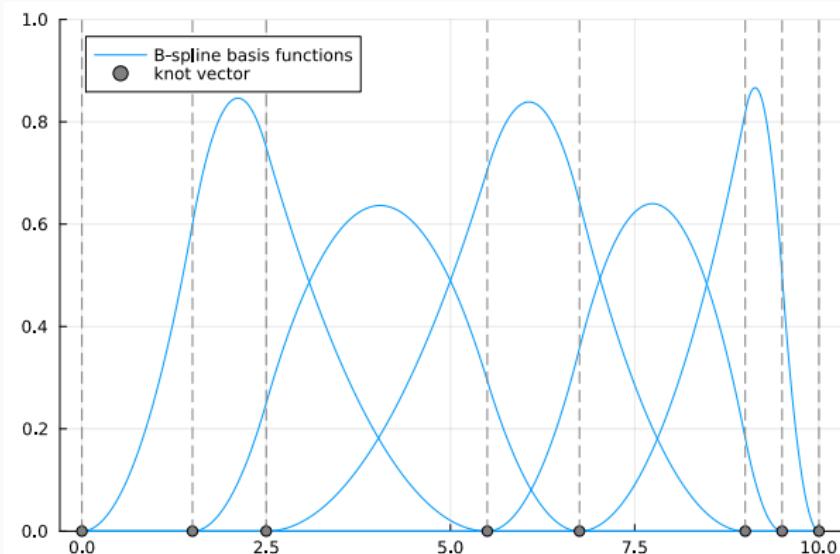
## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



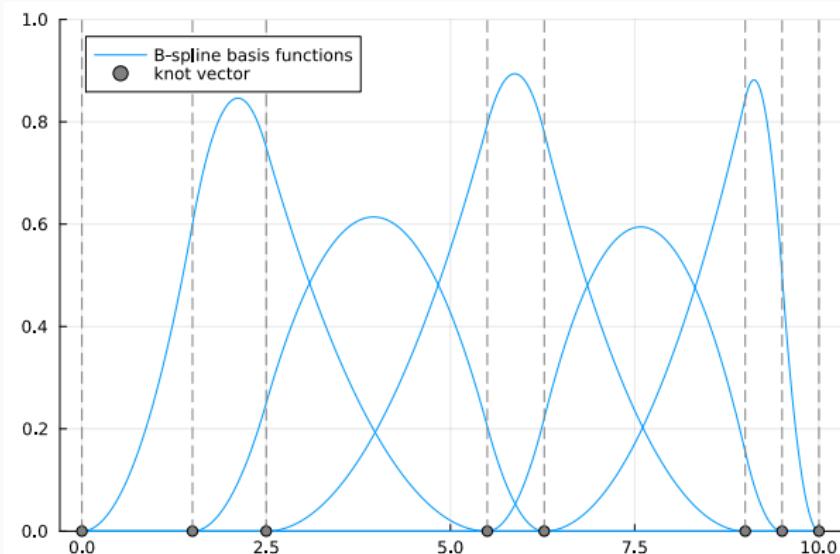
## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



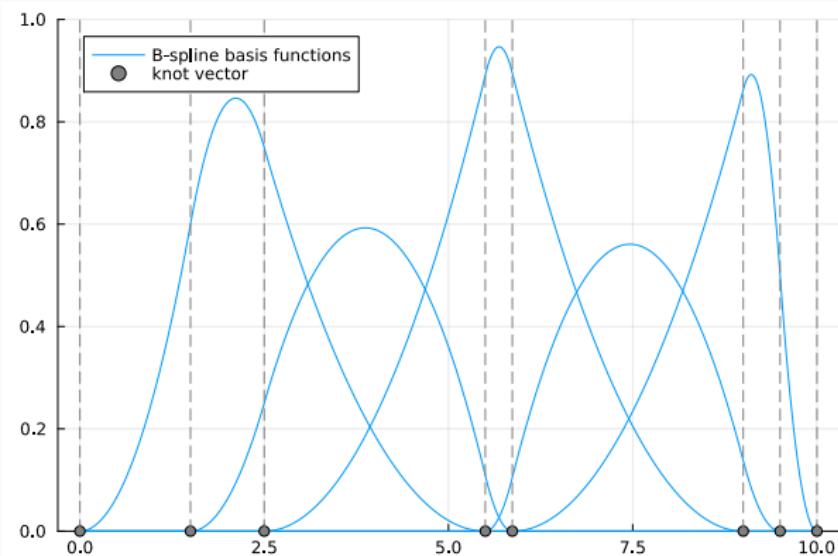
## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



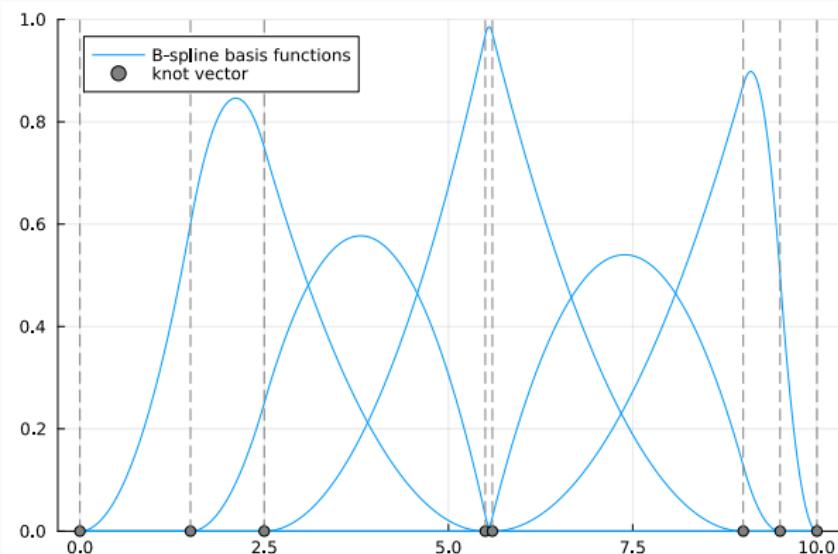
## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



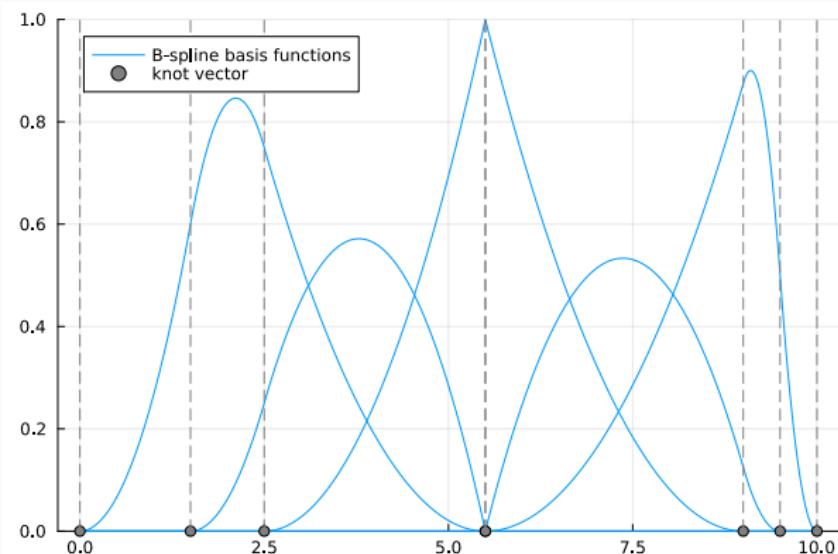
## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



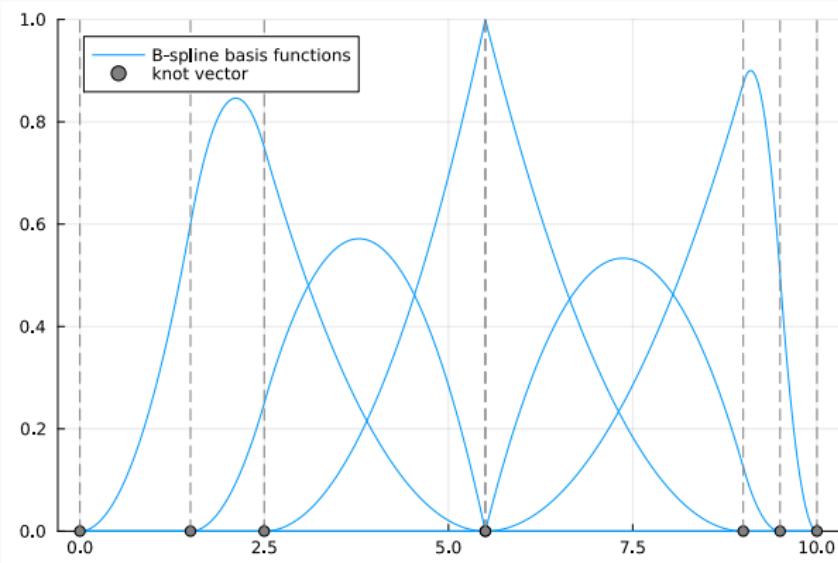
## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



## B-spline 基底関数の性質 ②

ノット列が重なるとどうなるか？



- 各点収束している
- 収束先で滑らかさが落ちている

## B-spline 基底関数の定義 (ふたたび)

ノット列の重複があるとき：

- 各点収束している
- 収束先で滑らかさが落ちている

しかし、B-spline 基底関数の定義はノット列の重複を許さないはず

$$B_{(i,p,k)}(t) = \frac{t - k_i}{k_{i+p} - k_i} B_{(i,p-1,k)}(t) + \frac{k_{i+p+1} - t}{k_{i+p+1} - k_{i+1}} B_{(i+1,p-1,k)}(t)$$
$$B_{(i,0,k)}(t) = \begin{cases} 1 & (k_i \leq t < k_{i+1}) \\ 0 & (\text{otherwise}) \end{cases}$$

## B-spline 基底関数の定義 (ふたたび)

ノット列の重複があるとき：

- 各点収束している
- 収束先で滑らかさが落ちている

しかし、B-spline 基底関数の定義はノット列の重複を許さないはず

$$B_{(i,p,k)}(t) = \frac{t - k_i}{k_{i+p} - k_i} B_{(i,p-1,k)}(t) + \frac{k_{i+p+1} - t}{k_{i+p+1} - k_{i+1}} B_{(i+1,p-1,k)}(t)$$
$$B_{(i,0,k)}(t) = \begin{cases} 1 & (k_i \leq t < k_{i+1}) \\ 0 & (\text{otherwise}) \end{cases}$$

ゼロ除算をゼロとすれば、収束先の関数が得られる！

区分多項式空間  $\mathcal{P}[p, k]$  の定義も拡張される

$$\mathcal{P}[p, k] = \operatorname{span}_i \{ B_{(i,p,k)} \} = \left\{ f : \mathbb{R} \rightarrow \mathbb{R} \middle| \begin{array}{l} \forall i, f|_{(k_i, k_{i+1})} \text{ は } p \text{ 次多項式} \\ \text{supp}(f) \subseteq [k_1, k_l] \\ f \text{ は } t \text{ の近傍で } C^{p-\mathfrak{n}_k(t)} \text{ 級} \end{array} \right\}$$

$$\mathfrak{n}_k(t) = \# \{ i \mid t = k_i \}$$

## これまでのあらすじ：

- (1) 与えられた関数  $f : I \rightarrow \mathbb{R}$  を区分多項式で近似したい ( $I$ : 有界閉区間)
- (2) B-spline 基底関数  $B_{(i,p,k)}$  の線型結合で近似するのが良さそう

$$f(t) \approx \sum_i a_i B_{(i,p,k)}(t)$$

- (3) ノット列  $k$  に重複があると  $B_{(i,p,k)}$  の滑らかさが落ちる
- (4)  $B_{(i,p,k)}$  の張る関数空間が  $\mathcal{P}[p, k]$

## 次に気になること：

- 関数だけでなく、曲線  $f : I \rightarrow V$  を近似できるか？

係数  $a_i \in \mathbb{R}$  を線形空間の元  $a_i \in V$  に置き換える

$$f(t) \approx \sum_i a_i B_{(i,p,k)}(t)$$

- 1変数関数だけでなく、2変数関数  $f : I \times J \rightarrow \mathbb{R}$  を近似できるか？  
B-spline 基底関数のテンソル積

$$f(u^1, u^2) \approx \sum_{i,j} a_{ij} B_{(i,p^1,k^1)}(u^1) B_{(j,p^2,k^2)}(u^2)$$

## これまでのあらすじ：

- (1) 与えられた関数  $f : I \rightarrow \mathbb{R}$  を区分多項式で近似したい ( $I$ : 有界閉区間)
- (2) B-spline 基底関数  $B_{(i,p,k)}$  の線型結合で近似するのが良さそう

$$f(t) \approx \sum_i a_i B_{(i,p,k)}(t)$$

- (3) ノット列  $k$  に重複があると  $B_{(i,p,k)}$  の滑らかさが落ちる
- (4)  $B_{(i,p,k)}$  の張る関数空間が  $\mathcal{P}[p, k]$

## 次に気になること：

- 関数だけでなく、曲線  $f : I \rightarrow V$  を近似できるか？

係数  $a_i \in \mathbb{R}$  を線形空間の元  $a_i \in V$  に置き換える

$$f(t) \approx \sum_i a_i B_{(i,p,k)}(t)$$

- 1変数関数だけでなく、2変数関数  $f : I \times J \rightarrow \mathbb{R}$  を近似できるか？

B-spline 基底関数のテンソル積

$$f(u^1, u^2) \approx \sum_{i,j} a_{ij} B_{(i,p^1,k^1)}(u^1) B_{(j,p^2,k^2)}(u^2)$$

## これまでのあらすじ：

- (1) 与えられた関数  $f : I \rightarrow \mathbb{R}$  を区分多項式で近似したい ( $I$ : 有界閉区間)
- (2) B-spline 基底関数  $B_{(i,p,k)}$  の線型結合で近似するのが良さそう

$$f(t) \approx \sum_i a_i B_{(i,p,k)}(t)$$

- (3) ノット列  $k$  に重複があると  $B_{(i,p,k)}$  の滑らかさが落ちる
- (4)  $B_{(i,p,k)}$  の張る関数空間が  $\mathcal{P}[p, k]$

## 次に気になること：

- 関数だけでなく、曲線  $f : I \rightarrow V$  を近似できるか？  
係数  $a_i \in \mathbb{R}$  を線形空間の元  $a_i \in V$  に置き換える

$$f(t) \approx \sum_i a_i B_{(i,p,k)}(t)$$

- 1変数関数だけでなく、2変数関数  $f : I \times J \rightarrow \mathbb{R}$  を近似できるか？  
B-spline 基底関数のテンソル積

$$f(u^1, u^2) \approx \sum_{i,j} a_{ij} B_{(i,p^1,k^1)}(u^1) B_{(j,p^2,k^2)}(u^2)$$

# B-spline 曲線 ①

## 定義 (B-spline 曲線)

$V$  を  $\mathbb{R}$ -線形空間、 $a_i$  を  $V$  上の点とする。このとき

$$p(t) = \sum_i a_i B_{(i,p,k)}(t) \quad (t \in [k_{1+p}, k_{l-p}])$$

で表される曲線を B-spline 曲線と呼ぶ。 $a_i$  を制御点と呼ぶ。

```
using StaticArrays
# ノット列の定義
k = KnotVector([0,1,2,3,3,4,5,6,7])
# 区分多項式空間の定義
P = BSplineSpace{2}(k)
n = dim(P)
# 制御点の定義
a = [SVector(cos(i),sin(i)) for i in 0:n-1]
# B-spline 曲線の定義
C = BSplineManifold(a, P)
plot(C)
```

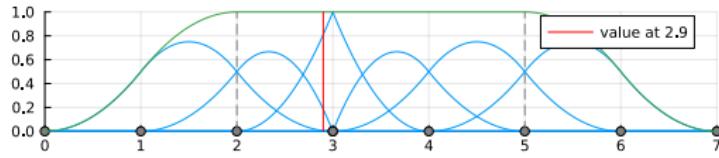
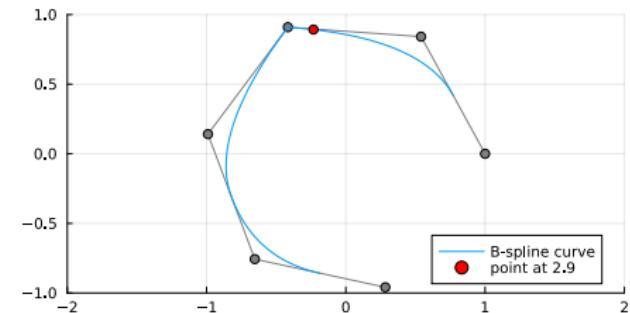
## B-spline 曲線 ②

### 定義 (B-spline 曲線)

$V$  を  $\mathbb{R}$ -線形空間、 $a_i$  を  $V$  上の点とする。このとき

$$p(t) = \sum_i a_i B_{(i,p,k)}(t) \quad (t \in [k_{1+p}, k_{l-p}])$$

で表される曲線を B-spline 曲線と呼ぶ。 $a_i$  を制御点と呼ぶ。



端点を通るとは限らない！

## B-spline 曲線 ③

制御点の平行移動で B-spline 曲線も平行移動する

```
a2 = [p+SVector(1,2) for p in a]
C2 = BSplineManifold(a2, P)
plot!(C2)
```

より一般に次の命題が成立

### 命題 (Affine 可換性)

$f : x \mapsto Ax + b$  を Affine 変換とし、制御点  $\{a_i\}$  から構成される B-spline 曲線を  $p(\{a_i\}; t)$  で表すとする。このとき次が成立。

$$p(\{f(a_i)\}; t) = f(p(\{a_i\}; t))$$

証明. 1 の分割の性質を使う

$$\begin{aligned} p(\{f(a_i)\}; t) &= \sum_i (Aa_i + b) B_{(i,p,k)}(t) = A \sum_i a_i B_{(i,p,k)}(t) + b \sum_i B_{(i,p,k)}(t) \\ &= A \sum_i a_i B_{(i,p,k)}(t) + b = f(p(\{a_i\}; t)) \end{aligned}$$

□

# B-spline 曲面

## 定義 (B-spline 曲面)

$V$  を  $\mathbb{R}$ -線形空間、 $a_{ij}$  を  $V$  上の点とする。このとき

$$\begin{aligned} p(u^1, u^2) &= \sum_{i,j} a_{ij} B_{(i,p^1,k^1)}(u^1) B_{(j,p^2,k^2)}(u^2) \\ ((u^1, u^2) &\in [k_{1+p^1}^1, k_{l^1-p^1}^1] \times [k_{1+p^2}^2, k_{l^2-p^2}^2]) \end{aligned}$$

で表される曲面を B-spline 曲面と呼ぶ。 $a_{ij}$  を制御点と呼ぶ。

```
# 区分多項式空間の定義
P = BSplineSpace{3}(KnotVector(0:7))
# 制御点の定義
a = [SVector(i-1.5, j-1.5, 1≤i≤2&&1≤j≤2)
      for i in 0:3, j in 0:3]
# B-spline 曲線の定義
M = BSplineManifold(a, P, P)
plotly()
plot(M)
gr()
```

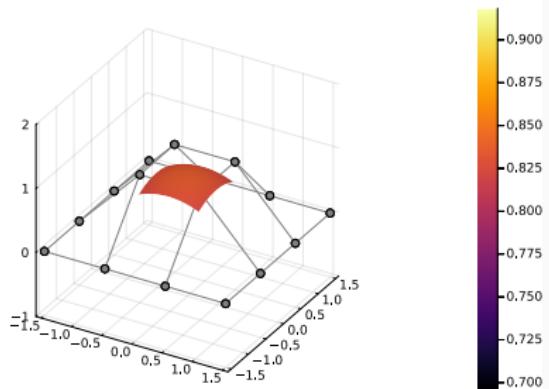
# B-spline 曲面

## 定義 (B-spline 曲面)

$V$  を  $\mathbb{R}$ -線形空間、 $a_{ij}$  を  $V$  上の点とする。このとき

$$\begin{aligned} p(u^1, u^2) &= \sum_{i,j} a_{ij} B_{(i,p^1,k^1)}(u^1) B_{(j,p^2,k^2)}(u^2) \\ ((u^1, u^2) &\in [k_{1+p^1}^1, k_{l^1-p^1}^1] \times [k_{1+p^2}^2, k_{l^2-p^2}^2]) \end{aligned}$$

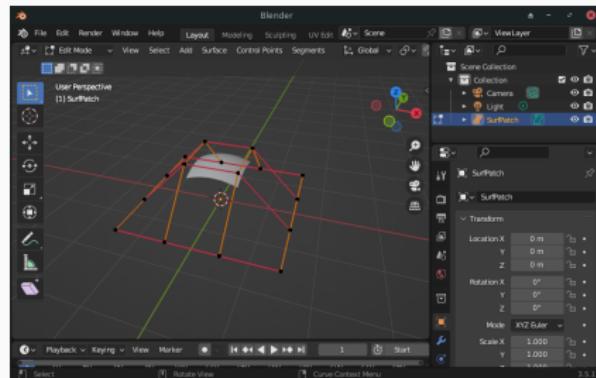
で表される曲面を B-spline 曲面と呼ぶ。 $a_{ij}$  を制御点と呼ぶ。



3 次元以上も同様に定義可能

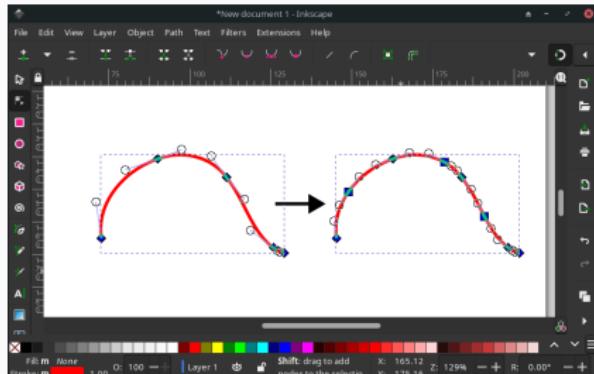
# GUI アプリケーションでの例

Blender



B-spline 曲面を編集可能

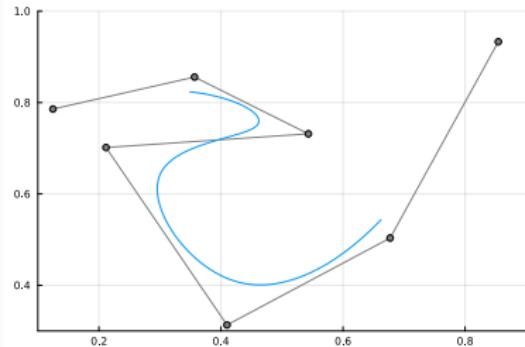
Inkscape



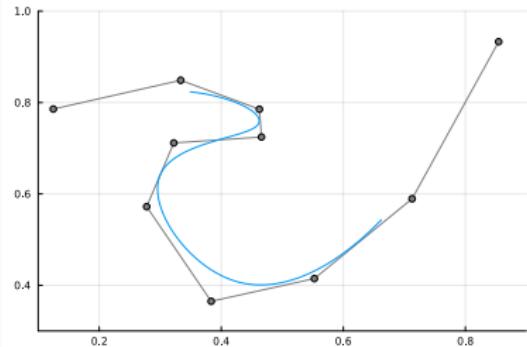
Bézier 曲線を編集可能

# リファインメント

制御点の数を増やす操作のこと



元の B-spline 曲線 ( $\mathcal{P}[p, k]$ )



リファインメント後 ( $\mathcal{P}[p', k']$ )

- (1) 新しい次数  $p'$  とノット列  $k'$  を決める
- (2) 次数とノット列に合わせて制御点  $a'_j$  を配置

$$\mathbf{p}(t) = \sum_{i=1}^n \mathbf{a}_i B_{(i,p,k)}(t) = \sum_{j=1}^{n'} \mathbf{a}'_j B_{(j,p',k')}(t)$$

$$B_{(i,p,k)} = \sum_{j=1}^{n'} A_{ij} B_{(j,p',k')} \quad \mathbf{a}'_j = \sum_{i=1}^n \mathbf{a}_i A_{ij}$$

# 区分多項式空間の包含関係

$B_{(i,p,k)} = \sum_j A_{ij} B_{(j,p',k')}$  を満たす行列  $A$  が存在するには  $\mathcal{P}[p, k] \subseteq \mathcal{P}[p', k']$  でなければならない

## 定理 (区分多項式空間の包含関係)

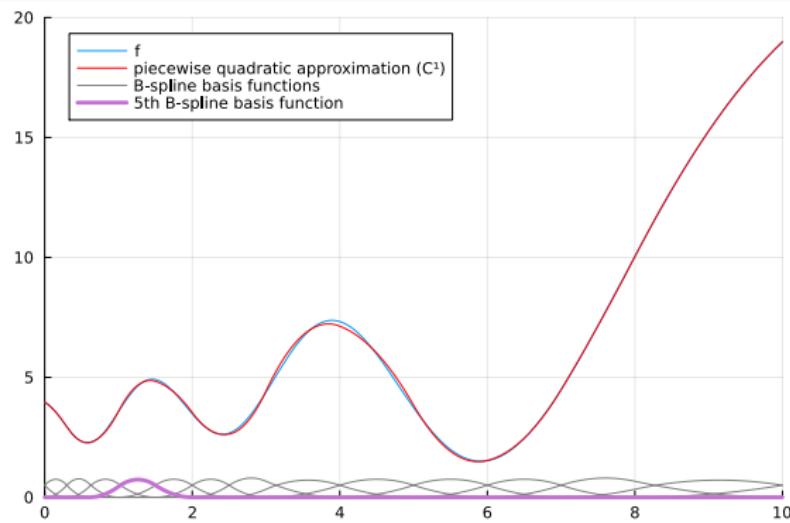
$$\mathcal{P}[p, k] \subseteq \mathcal{P}[p, k'] \Leftrightarrow k \subseteq k'$$

$p < p'$  の場合はもう少し複雑な条件になる

```
k = KnotVector(0:10)
k' = k + KnotVector([3.4, 5.0, 6.3])
P = BSplineSpace{3}(k)
P' = BSplineSpace{3}(k')
k ⊆ k'
P ⊆ P'
plot(P)
plot!(P')
A = changebasis(P, P')
```

1. 自己紹介
2. B-spline 最速入門
3. B-spline もう少し入門
4. B-spline の応用例
  - 関数近似
  - 滑らかなグラフを出力する
  - 内挿補間
  - 数値計算の例：編み紙
5. まとめ

# 関数を近似するには？(伏線回収)



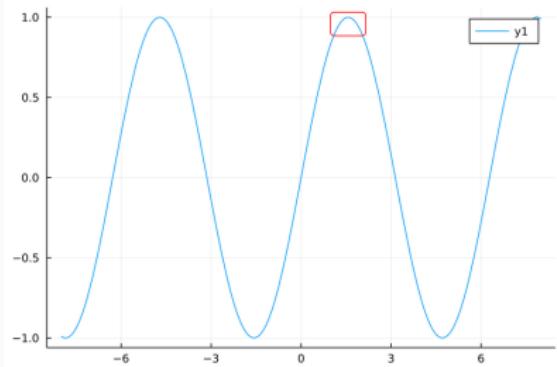
最小2乗法で次式を最小化して係数  $\{a_i\}$  を決めれば良い

$$\int_I \left( f(t) - \sum_i a_i B_{(i,p,k)}(t) \right)^2 dt$$

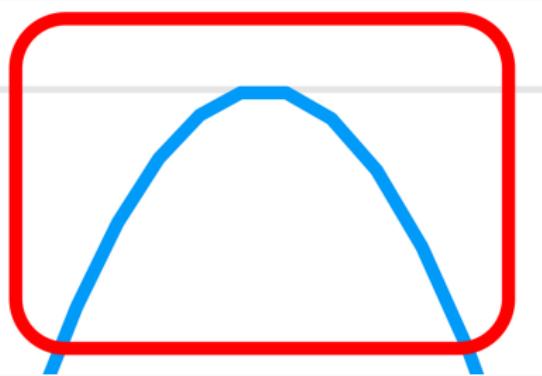
```
using BasicBSplineFitting
f(x) = (1+x^2/10)*sin(exp(-(x-12)/4))+x/2+3
a = fittingcontrolpoints_I(f, P)
```

# 滑らかなグラフを出力するには？①

Plots.jl の出力グラフは折れ線



`plot(sin, -8, 8)` の出力



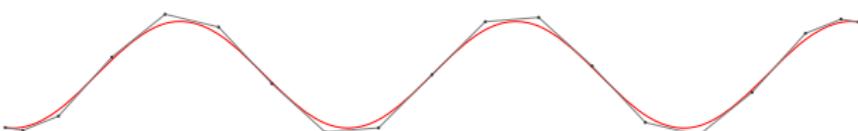
拡大

滑らかなグラフが欲しい！

## 滑らかなグラフを出力するには？②

```
using BasicBSpline
using BasicBSplineFitting
using BasicBSplineExporter
using StaticArrays
f(t) = SVector(t,sin(t))
a, b = -8, 8
p = 3
k = KnotVector(a:b)+p*KnotVector([a,b])
P = BSplineSpace{p}(k)
a = fittingcontrolpoints_I(f,(P,))
C = BSplineManifold(a,(P,))
save_svg("sin.svg", C, xlims=(-10,10), ylims=(-2,2))
```

BasicBSplineExporter.jl パッケージで画像保存が可能！

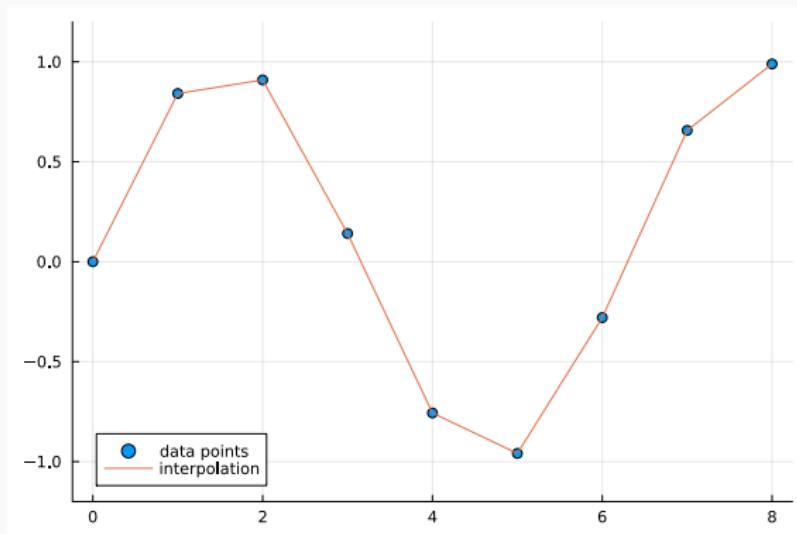


# 点列を内挿補間するには？①

以下の条件の補間を考える

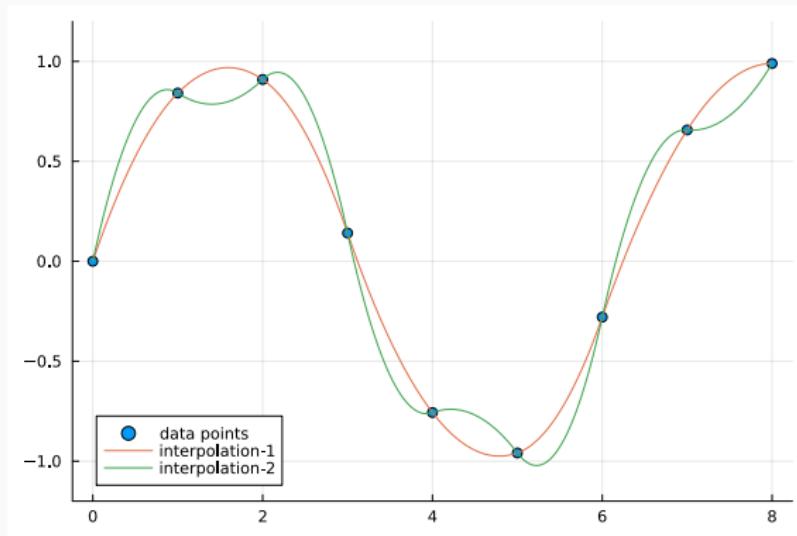
- $p$  次区分多項式で内挿補間
- 区分点とデータ点が一致
- 区分点の近傍で  $C^{p-1}$  級

1次区分多項式では自明



## 点列を内挿補間するには？②

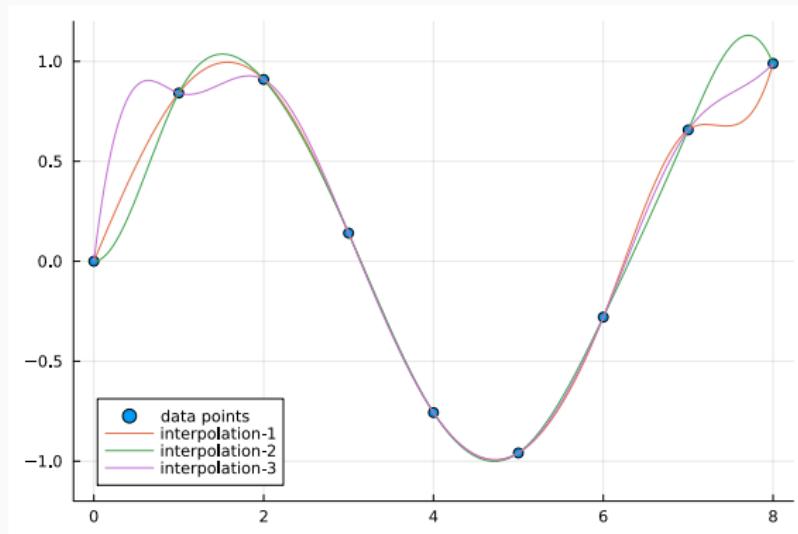
2次区分多項式では一意的ではない



データ点を通る区分多項式は無数に存在 (1自由度)

## 点列を内挿補間するには？③

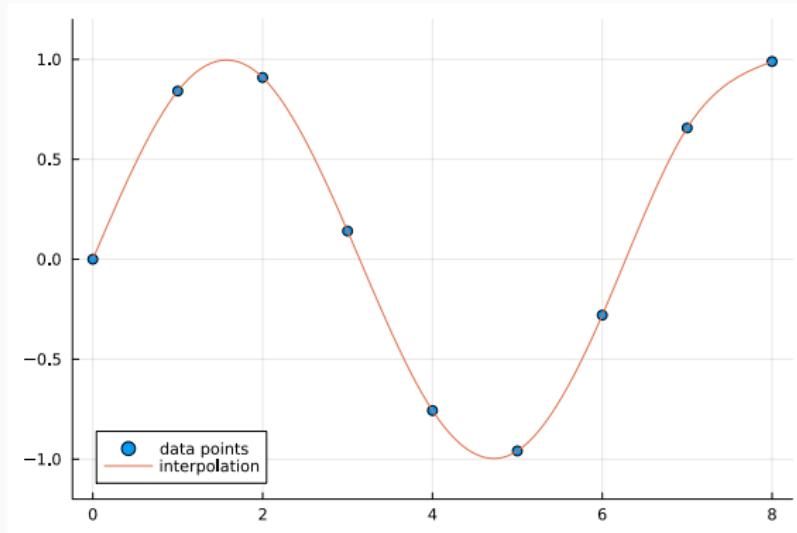
3次区分多項式でも一意的ではない



データ点を通る区分多項式は無数に存在 (2自由度)

## 点列を内挿補間するには？④

3次区分多項式でも一意的ではない



データ点を通る区分多項式は無数に存在 (2自由度)

両端での2階微分を0とすれば一意的！

## 点列を内挿補間するには？⑤

BasicBSpline.jl では内挿補間の API は無いが、実装は容易

```
function interpolate(xs, fs::AbstractVector{T}) where T
    k = KnotVector(xs) + KnotVector([xs[1], xs[end]]) * 3
    P = BSplineSpace{3}(k)
    ddP = BSplineDerivativeSpace{2}(P)
    dda = [bsplinebasis(ddP, j, xs[1]) for j in 1:dim(P)]
    ddb = [bsplinebasis(ddP, j, xs[end]) for j in 1:dim(P)]
    M = [bsplinebasis(P, j, x) for x in xs, j in 1:dim(P)]
    M = vcat(dda', M, ddb')
    y = vcat(zero(T), fs, zero(T))
    return BSplineManifold(M\y, P)
end
ts = 0:8
F = interpolate(ts, sin.(ts))
scatter(ts, sin.(ts), label="data points")
plot!(t->F(t), 0, 8, label="interpolation")
```

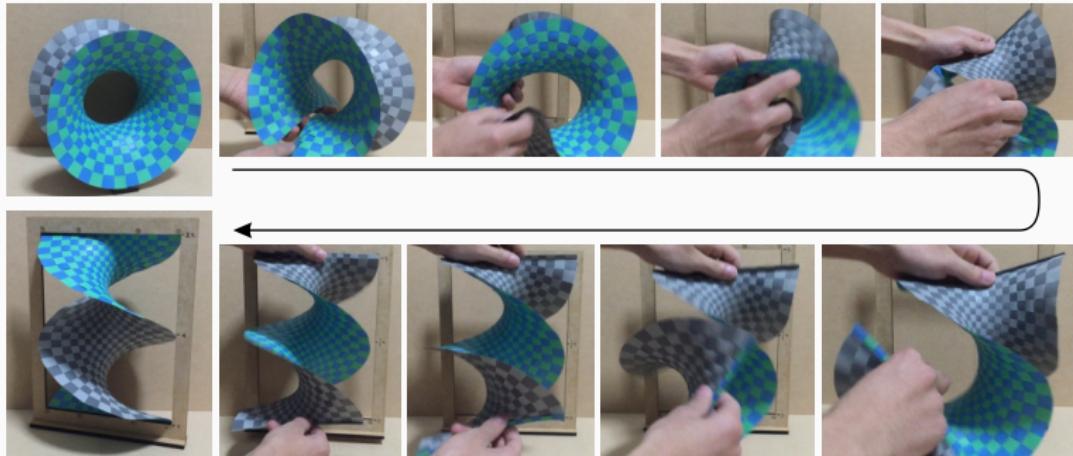
## 紙を編んで滑らかな曲面を作る話



埋め込み形状 (上図中央) を B-spline 多様体として数値計算

- 曲面片の弾性変形エネルギーを最小化 (偏微分方程式)
- 制御点を未知変数として Galerkin 法が使える
- B-spline によって滑らかな ( $C^2$  級の) 解が得られる
- 3 次 B-spline 曲線は SVG 画像として出力可能

本日、曲面模型を持ってきています！



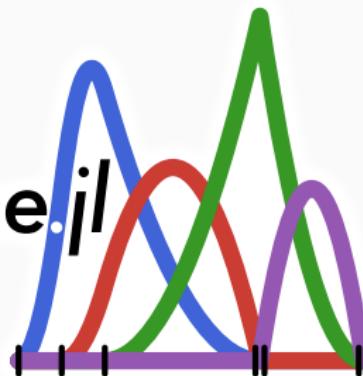
詳細は JuliaCon2023 にて！

1. 自己紹介
2. B-spline 最速入門
3. B-spline もう少し入門
4. B-spline の応用例
5. まとめ

# まとめ

- B-spline は滑らかな区分多項式を扱うための道具  
「区分関数」と「多項式」両方の性質を持つ ♠
- B-spline は数学的に奥深い  
区分多項式空間  $\mathcal{P}[p, k]$  の性質、など
- B-spline は応用先が広い  
幾何形状表現、内挿補間、Galerkin 法など
- B-spline を Julia で扱うには BasicBSpline.jl が便利！  
あなたの Star★をお待ちしています！

*BasicBSpline.jl*



計算が速い、以外にも色々ある

- Unicode コーディング

```
k1 ⊑ k2, k1 + k2, P1 ⊑ P2
```

- 抽象型・具象型

```
KnotVector <: AbstractKnotVector,
```

```
UniformKnotVector <: AbstractKnotVector
```

- パラメトリック型

```
BSplineSpace{p}
```

- メタプログラミング

```
P = BSplineSpace{3}(KnotVector(1:8))
```

```
@less bsplinebasisall(P, 1, 4.2)
```

## 他パッケージとの比較

- sostock/BSplines.jl ★22
  - B-spline の基本的な操作を実装
  - 多項式次数  $p$  が型パラメータに含まれないので遅い
- jipolanco/BSplineKit.jl ★40
  - 応用を重視した B-spline パッケージ
  - 悪い歴史的経緯を踏襲している (e.g. `BSplineOrder`)
- HoBeZwe/NURBS.jl ★2
  - 幾何形状表現への応用を意図したパッケージ
  - 実装途中の箇所が多い
- kbarbary/Dierckx.jl ★141
  - Fortran ライブラリの DIERCKX のラッパー
  - Pure Julia ではない (e.g. 有理数型が使えない)
- **hyrodium/BasicBSpline.jl** ★76
  - 数学志向で高速な B-spline の実装

(公平な比較になっていないことに注意)

- Elaine Cohen, Richard F. Riesenfeld, Gershon Elber (2001) “Geometric Modeling with Splines: An Introduction”
- Larry Schumaker (1989) “Spline Functions: Basic Theory”
- 市田浩三, 吉本富士市 (1979) 『スプライン関数とその応用』