Adaptieve benaderingsmethodes

Jan Westerdiep

8 april 2014

Wat?

- Adaptieve benaderingsmethodes

Wat?

- Adaptieve benaderingsmethodes
- Benaderen dmv. (stuksgewijs) polynomen

Wat?

- Adaptieve benaderingsmethodes
- Benaderen dmv. (stuksgewijs) polynomen
- Door in te zoomen daar waar het probleem moeilijk is

Wat?

- Adaptieve benaderingsmethodes
- Benaderen dmv. (stuksgewijs) polynomen
- Door in te zoomen daar waar het probleem moeilijk is
- Dit alles adhv. 2 algoritmes van Peter Binev (2004, 2013)

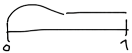
Wat?

- Adaptieve benaderingsmethodes
- Benaderen dmv. (stuksgewijs) polynomen
- Door in te zoomen daar waar het probleem moeilijk is
- Dit alles adhv. 2 algoritmes van Peter Binev (2004, 2013)
- Begeleiding door:
  1. Wiskunde – Rob Stevenson
  2. Informatica – Dick van Albada

h-verfijnend algoritme

- Laat n vast, $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$

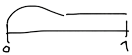
f:



h-verfijnend algoritme

- Laat n vast, $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:

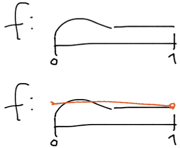
f:



h-verfijnend algoritme

- Laat n vast, $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:
  1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)} \quad \text{(Stelling: dit kan)}$$

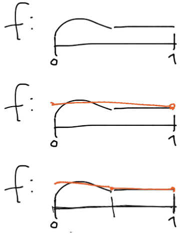


h-verfijnend algoritme

- Laat n vast, $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:
  1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)} \quad \text{(Stelling: dit kan)}$$

  2. Kies $\Delta \in P$ en hak deze in twee (subdivision)

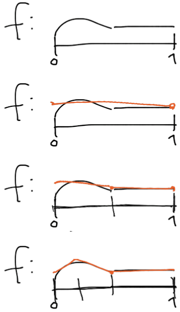


h-verfijnend algoritme

- Laat n vast, $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:
  1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)} \quad \text{(Stelling: dit kan)}$$

  2. Kies $\Delta \in P$ en hak deze in twee (subdivision)
  3. Herhaal

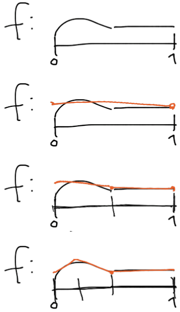


h-verfijnend algoritme

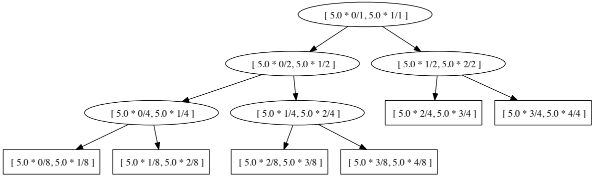
- Laat n vast, $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:
  1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)} \quad \text{(Stelling: dit kan)}$$

  2. Kies $\Delta \in P$ en hak deze in twee (subdivision)
  3. Herhaal



$\Rightarrow$ Tree-Generating Algoritme



h-verfijnende algoritmes

1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)}$$

2. Kies $\Delta \in P$ en hak deze in twee (subdivision)

h-verfijnende algoritmes

1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)}$$

2. Kies $\Delta \in P$ en hak deze in twee (subdivision)

- *Greedy*: kies argmax$\{e(\Delta)\}$

h-verfijnende algoritmes

1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)}$$

2. Kies $\Delta \in P$ en hak deze in twee (subdivision)

- *Greedy*: kies $\text{argmax}\{e(\Delta)\}$
- *Binev-2004*: maak een slimmere keuze,

h-verfijnende algoritmes

1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)}$$

2. Kies $\Delta \in P$ en hak deze in twee (subdivision)

- *Greedy*: kies $\operatorname{argmax}\{e(\Delta)\}$
- *Binev-2004*: maak een slimmere keuze, bewijsbaar:

$\Rightarrow$ Binev-2004 maakt een boom die na n opdelingen

h-verfijnende algoritmes

1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)}$$

2. Kies $\Delta \in P$ en hak deze in twee (subdivision)

- *Greedy*: kies $\operatorname{argmax}\{e(\Delta)\}$
- *Binev-2004*: maak een slimmere keuze, bewijsbaar:

$\Rightarrow$ Binev-2004 maakt een boom die na n opdelingen

- gevonden is in $\mathcal{O}(n)$ operaties,

h-verfijnende algoritmes

1. Voor elke $\Delta \in P$, bepaal

$$e(\Delta) := \inf_{q \in \mathcal{P}^n} \|f - q\|^2_{L^2(\Delta)}$$

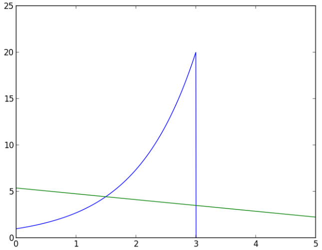
2. Kies $\Delta \in P$ en hak deze in twee (subdivision)

- *Greedy*: kies $\operatorname{argmax}\{e(\Delta)\}$
- *Binev-2004*: maak een slimmere keuze, bewijsbaar:

$\Rightarrow$ Binev-2004 maakt een boom die na n opdelingen

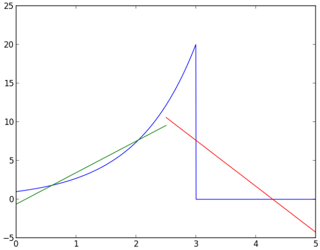
- gevonden is in $\mathcal{O}(n)$ operaties, en
- beter is dan de optimale boom in $n/2$ opdelingen

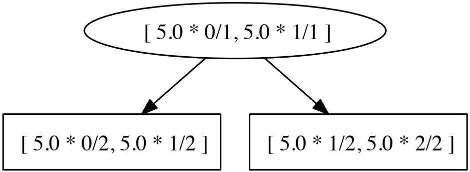
Binev-2004



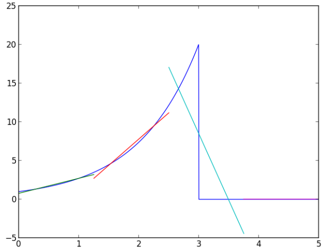
[5.0 * 0/1 , 5.0 * 1/1]

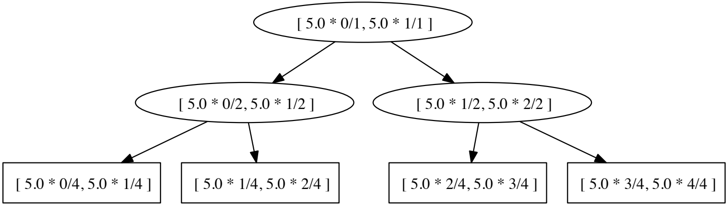
Binev-2004



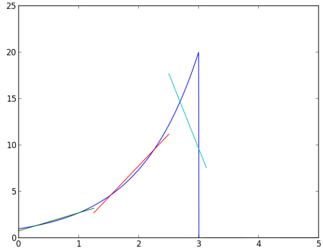


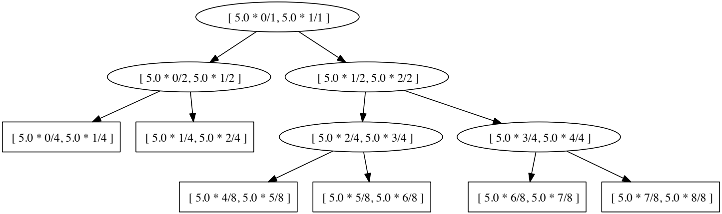
Binev-2004


25
20
15
10
5
0
−5
0
1
2
3
4
5

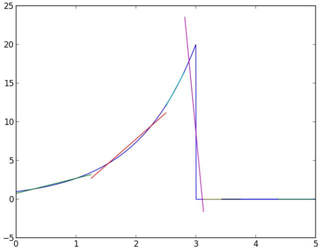

[5.0 * 0/1, 5.0 * 1/1]
[5.0 * 0/2, 5.0 * 1/2]
[5.0 * 1/2, 5.0 * 2/2]
[5.0 * 0/4, 5.0 * 1/4]
[5.0 * 1/4, 5.0 * 2/4]
[5.0 * 2/4, 5.0 * 3/4]
[5.0 * 3/4, 5.0 * 4/4]

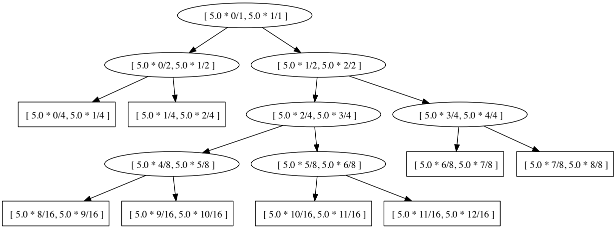
Binev-2004


25
20
15
10
5
0
0
1
2
3
4
5

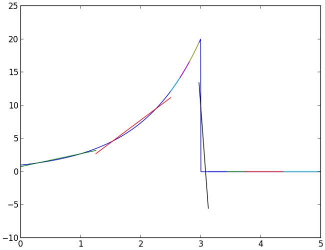

[5.0 * 0/1, 5.0 * 1/1]
[5.0 * 0/2, 5.0 * 1/2]
[5.0 * 1/2, 5.0 * 2/2]
[5.0 * 0/4, 5.0 * 1/4]
[5.0 * 1/4, 5.0 * 2/4]
[5.0 * 2/4, 5.0 * 3/4]
[5.0 * 3/4, 5.0 * 4/4]
[5.0 * 4/8, 5.0 * 5/8]
[5.0 * 5/8, 5.0 * 6/8]
[5.0 * 6/8, 5.0 * 7/8]
[5.0 * 7/8, 5.0 * 8/8]

Binev-2004

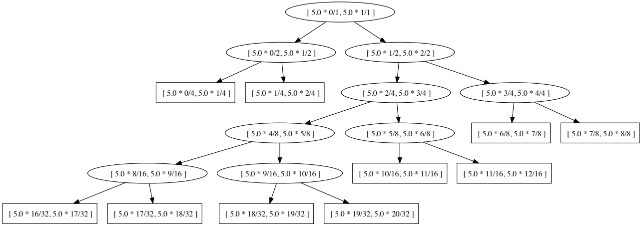

25
20
15
10
5
0
−5
0
1
2
3
4
5


[5.0 * 0/1, 5.0 * 1/1]
[5.0 * 0/2, 5.0 * 1/2]
[5.0 * 1/2, 5.0 * 2/2]
[5.0 * 0/4, 5.0 * 1/4]
[5.0 * 1/4, 5.0 * 2/4]
[5.0 * 2/4, 5.0 * 3/4]
[5.0 * 3/4, 5.0 * 4/4]
[5.0 * 4/8, 5.0 * 5/8]
[5.0 * 5/8, 5.0 * 6/8]
[5.0 * 6/8, 5.0 * 7/8]
[5.0 * 7/8, 5.0 * 8/8]
[5.0 * 8/16, 5.0 * 9/16]
[5.0 * 9/16, 5.0 * 10/16]
[5.0 * 10/16, 5.0 * 11/16]
[5.0 * 11/16, 5.0 * 12/16]

Binev-2004

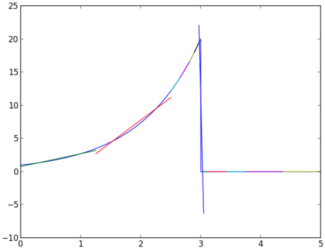


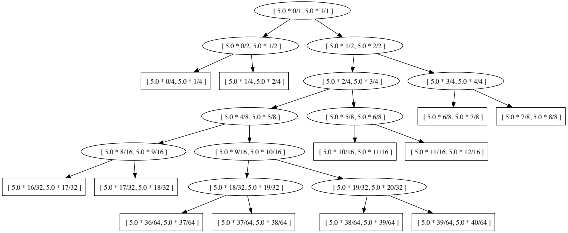
25
20
15
10
5
0
−5
−10
0
1
2
3
4
5



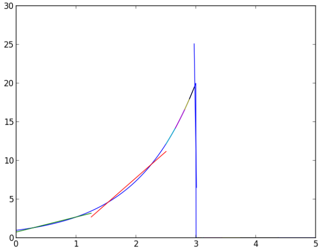
[5.0 * 0/1, 5.0 * 1/1]
[5.0 * 0/2, 5.0 * 1/2]
[5.0 * 1/2, 5.0 * 2/2]
[5.0 * 0/4, 5.0 * 1/4]
[5.0 * 1/4, 5.0 * 2/4]
[5.0 * 2/4, 5.0 * 3/4]
[5.0 * 3/4, 5.0 * 4/4]
[5.0 * 4/8, 5.0 * 5/8]
[5.0 * 5/8, 5.0 * 6/8]
[5.0 * 6/8, 5.0 * 7/8]
[5.0 * 7/8, 5.0 * 8/8]
[5.0 * 8/16, 5.0 * 9/16]
[5.0 * 9/16, 5.0 * 10/16]
[5.0 * 10/16, 5.0 * 11/16]
[5.0 * 11/16, 5.0 * 12/16]
[5.0 * 16/32, 5.0 * 17/32]
[5.0 * 17/32, 5.0 * 18/32]
[5.0 * 18/32, 5.0 * 19/32]
[5.0 * 19/32, 5.0 * 20/32]

Binev-2004

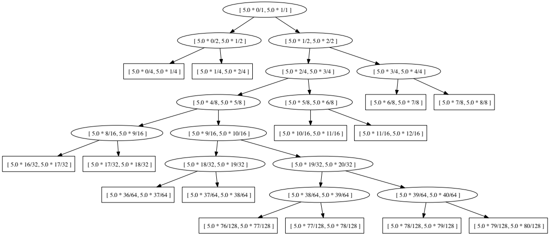

25
20
15
10
5
0
−5
−10
0
1
2
3
4
5


[5.0 * 0/1, 5.0 * 1/1]
[5.0 * 0/2, 5.0 * 1/2]
[5.0 * 1/2, 5.0 * 2/2]
[5.0 * 0/4, 5.0 * 1/4]
[5.0 * 1/4, 5.0 * 2/4]
[5.0 * 2/4, 5.0 * 3/4]
[5.0 * 3/4, 5.0 * 4/4]
[5.0 * 4/8, 5.0 * 5/8]
[5.0 * 5/8, 5.0 * 6/8]
[5.0 * 6/8, 5.0 * 7/8]
[5.0 * 7/8, 5.0 * 8/8]
[5.0 * 8/16, 5.0 * 9/16]
[5.0 * 9/16, 5.0 * 10/16]
[5.0 * 10/16, 5.0 * 11/16]
[5.0 * 11/16, 5.0 * 12/16]
[5.0 * 16/32, 5.0 * 17/32]
[5.0 * 17/32, 5.0 * 18/32]
[5.0 * 18/32, 5.0 * 19/32]
[5.0 * 19/32, 5.0 * 20/32]
[5.0 * 36/64, 5.0 * 37/64]
[5.0 * 37/64, 5.0 * 38/64]
[5.0 * 38/64, 5.0 * 39/64]
[5.0 * 39/64, 5.0 * 40/64]

Binev-2004



30
25
20
15
10
5
0
0
1
2
3
4
5

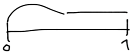


[5.0 * 0/1, 5.0 * 1/1]
[5.0 * 0/2, 5.0 * 1/2]
[5.0 * 1/2, 5.0 * 2/2]
[5.0 * 0/4, 5.0 * 1/4]
[5.0 * 1/4, 5.0 * 2/4]
[5.0 * 2/4, 5.0 * 3/4]
[5.0 * 3/4, 5.0 * 4/4]
[5.0 * 4/8, 5.0 * 5/8]
[5.0 * 5/8, 5.0 * 6/8]
[5.0 * 6/8, 5.0 * 7/8]
[5.0 * 7/8, 5.0 * 8/8]
[5.0 * 8/16, 5.0 * 9/16]
[5.0 * 9/16, 5.0 * 10/16]
[5.0 * 10/16, 5.0 * 11/16]
[5.0 * 11/16, 5.0 * 12/16]
[5.0 * 16/32, 5.0 * 17/32]
[5.0 * 17/32, 5.0 * 18/32]
[5.0 * 18/32, 5.0 * 19/32]
[5.0 * 19/32, 5.0 * 20/32]
[5.0 * 36/64, 5.0 * 37/64]
[5.0 * 37/64, 5.0 * 38/64]
[5.0 * 38/64, 5.0 * 39/64]
[5.0 * 39/64, 5.0 * 40/64]
[5.0 * 76/128, 5.0 * 77/128]
[5.0 * 77/128, 5.0 * 78/128]
[5.0 * 78/128, 5.0 * 79/128]
[5.0 * 79/128, 5.0 * 80/128]

hp-verfijnend algoritme

- Naast partitie verfijnen nu ook polynomiale graad verhogen

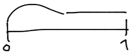
f:



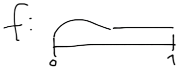
hp-verfijnend algoritme

- Naast partitie verfijnen nu ook polynomiale graad verhogen
- $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$

f:



hp-verfijnend algoritme

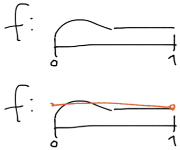


- Naast partitie verfijnen nu ook polynomiale graad verhogen
- $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:

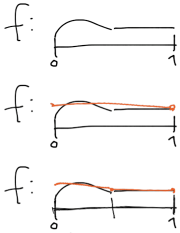
hp-verfijnend algoritme

- Naast partitie verfijnen nu ook polynomiale graad verhogen
- $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:
  1. Voor elke $\Delta \in P$, bepaal

$$e_{n(\Delta)}(\Delta) := \inf_{q \in \mathcal{P}^{n(\Delta)}} \|f - q\|^2_{L^2(\Delta)}$$



hp-verfijnend algoritme



- Naast partitie verfijnen nu ook polynomiale graad verhogen
- $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:
  1. Voor elke $\Delta \in P$, bepaal

$$e_{n(\Delta)}(\Delta) := \inf_{q \in \mathcal{P}^{n(\Delta)}} \|f - q\|^2_{L^2(\Delta)}$$

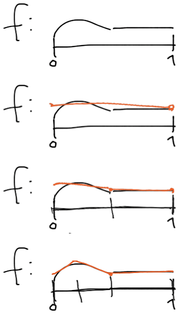
  2. Kies $\Delta \in P$ en hak deze in twee (subdivision)

hp-verfijnend algoritme

- Naast partitie verfijnen nu ook polynomiale graad verhogen
- $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:
  1. Voor elke $\Delta \in P$, bepaal

$$e_{n(\Delta)}(\Delta) := \inf_{q \in \mathcal{P}^{n(\Delta)}} \|f - q\|^2_{L^2(\Delta)}$$

  2. Kies $\Delta \in P$ en hak deze in twee (subdivision)
  3. Herhaal



hp-verfijnend algoritme

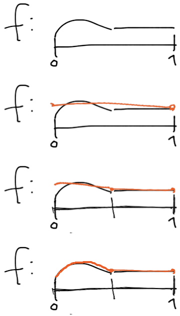
- Naast partitie verfijnen nu ook polynomiale graad verhogen
- $f : [a, b] \to \mathbb{R}$ met partitie $P = \{[a, b]\}$
- Partitie P adaptief verfijnen, dus itereer:
  1. Voor elke $\Delta \in P$, bepaal

$$e_{n(\Delta)}(\Delta) := \inf_{q \in \mathcal{P}^{n(\Delta)}} \|f - q\|^2_{L^2(\Delta)}$$

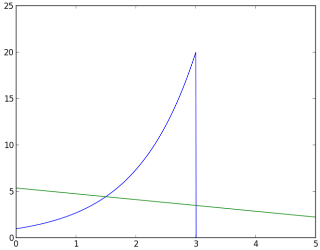
  2. Kies $\Delta \in P$ en hak deze in twee (subdivision)

  $\Rightarrow$ Of verhoog $n(\Delta)$

  3. Herhaal

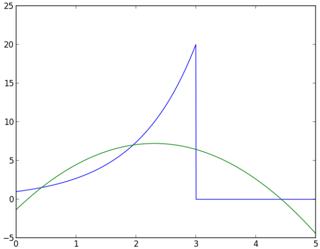


Binev-2013



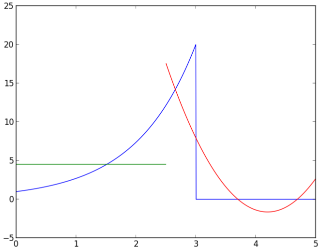
$$\boxed{\begin{array}{c}[\ 5.0 * 0/1, 5.0 * 1/1\ ] \\ \text{p=2}\end{array}}$$

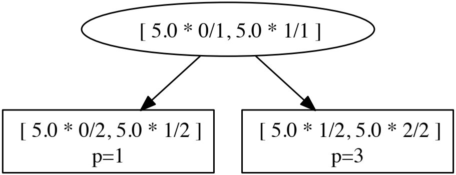
Binev-2013



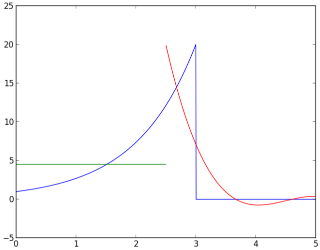
[5.0 * 0/1, 5.0 * 1/1]
p=3

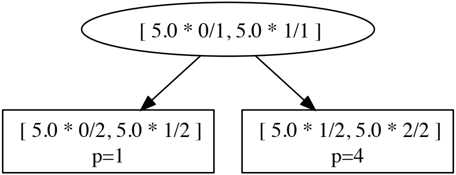
Binev-2013



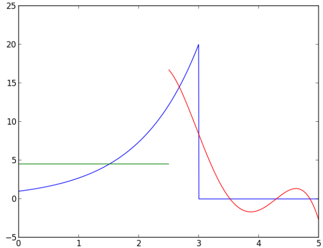


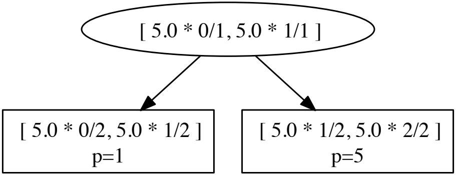
Binev-2013



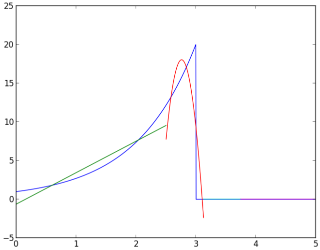


Binev-2013

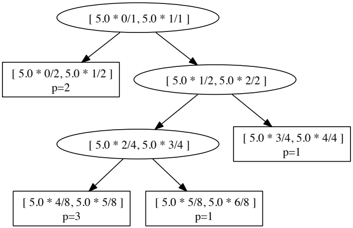




Binev-2013

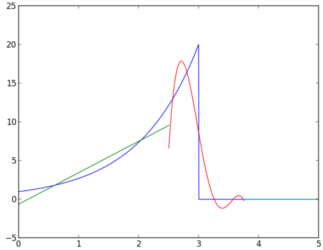


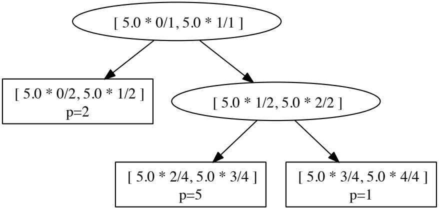
25
20
15
10
5
0
−5
0
1
2
3
4
5



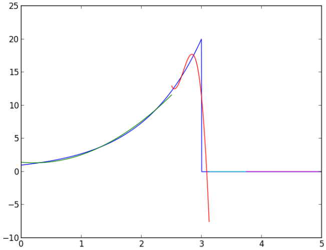
[5.0 * 0/1, 5.0 * 1/1]
[5.0 * 0/2, 5.0 * 1/2]
p=2
[5.0 * 1/2, 5.0 * 2/2]
[5.0 * 2/4, 5.0 * 3/4]
[5.0 * 3/4, 5.0 * 4/4]
p=1
[5.0 * 4/8, 5.0 * 5/8]
p=3
[5.0 * 5/8, 5.0 * 6/8]
p=1

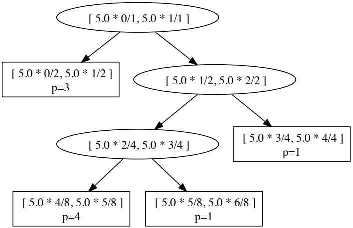
Binev-2013


25
20
15
10
5
0
−5
0
1
2
3
4
5

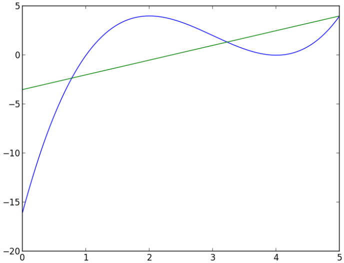

[5.0 * 0/1, 5.0 * 1/1]
[5.0 * 0/2, 5.0 * 1/2]
p=2
[5.0 * 1/2, 5.0 * 2/2]
[5.0 * 2/4, 5.0 * 3/4]
p=5
[5.0 * 3/4, 5.0 * 4/4]
p=1

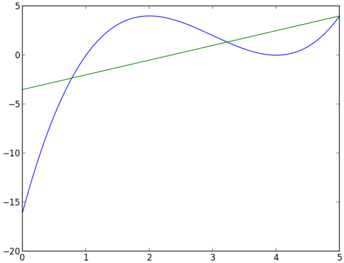
Binev-2013


25
20
15
10
5
0
−5
−10
0
1
2
3
4
5

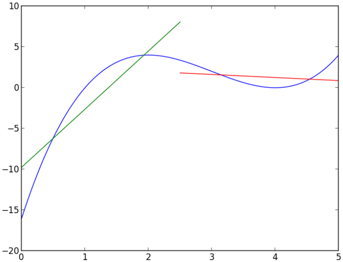

[5.0 * 0/1, 5.0 * 1/1]
[5.0 * 0/2, 5.0 * 1/2]
p=3
[5.0 * 1/2, 5.0 * 2/2]
[5.0 * 2/4, 5.0 * 3/4]
[5.0 * 3/4, 5.0 * 4/4]
p=1
[5.0 * 4/8, 5.0 * 5/8]
p=4
[5.0 * 5/8, 5.0 * 6/8]
p=1

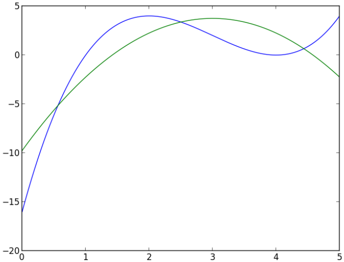
Ander voorbeeld



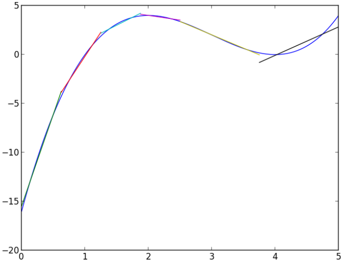


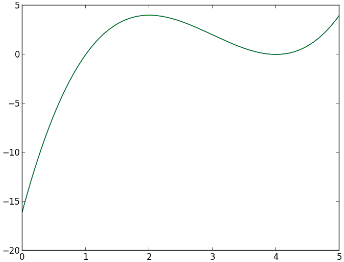
Ander voorbeeld



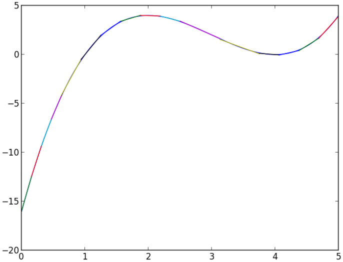


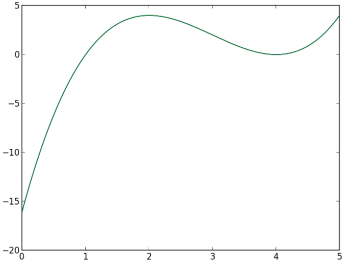
Ander voorbeeld





Ander voorbeeld





Wanneer?

✓ Feb, maart: theorie lezen, ± 6 pagina's tekst, implementatie Python

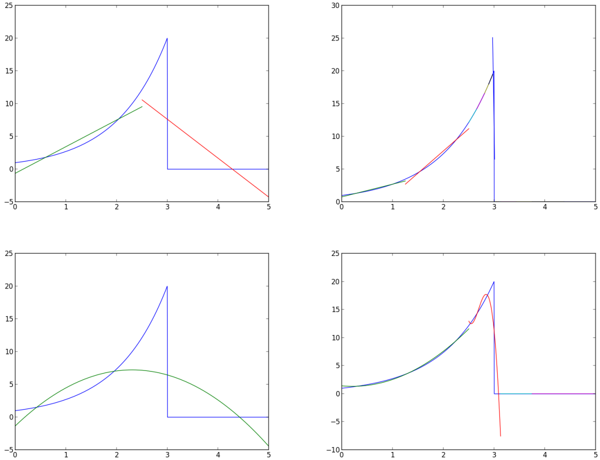
Wanneer?

- ✓ Feb, maart: theorie lezen, ± 6 pagina's tekst, implementatie Python
- April, mei: theorie op papier, implementatie in C

Wanneer?

- ✓ Feb, maart: theorie lezen, ± 6 pagina's tekst, implementatie Python
- April, mei: theorie op papier, implementatie in C
- Daarna: 2-dimensionaal, parallelliseren, toepassen

Jan Westerdiep – Adaptieve benaderingsmethodes



Schets van inhoudsopgave

Schets van inhoudsopgave

Schets van inhoudsopgave

Schets van inhoudsopgave

Schets van inhoudsopgave

1. 1-dimensionale geval
2. 2-dimensionale geval
3. Implementatie in C
4. Parallelliseren
5. Implementatie in e.o.a. parallelle situatie (GPU, supercomputer)

Schets van inhoudsopgave

1. 1-dimensionale geval
2. 2-dimensionale geval
3. Implementatie in C
4. Parallelliseren
5. Implementatie in e.o.a. parallelle situatie (GPU, supercomputer)
6. (wellicht) Uitbreiding en toepassing