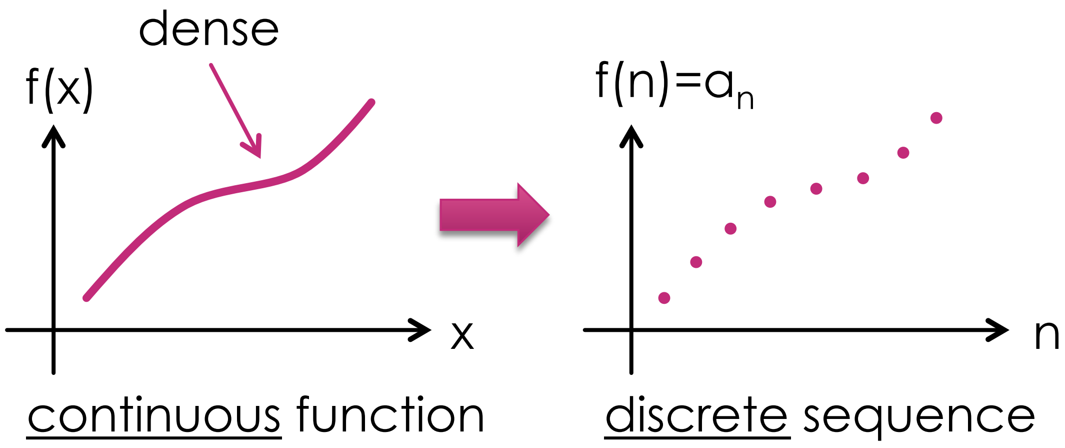
Sequences & Series

Sequences

✓ **A function of integer numbers**

- $\mathbf{a_n}$ = f(n): value of a_n can be **real** or **complex**
- Can be defined for positive and negative *n*
  - n = …, -3, -2, -1, 0, 1, 2, 3, …
- Example:
  - $a_n = 1/n^2$
- Special types of sequences
  - Monotonically increasing: $\mathbf{a_{n+1} > a_n}$ for all *n*
    - increasing: $a_{n+1} \geq a_n$ for all n
  - Monotonically decreasing: $\mathbf{a_{n+1} < a_n}$ for all *n*
    - Decreasing: $a_{n+1} \leq a_n$ for all n
- Limit of a sequence
  - Lim a_n for $n \rightarrow \infty$



$$\lim_{n\to\infty} a_n = \text{limit of a sequence}$$

$$\lim_{n\to\infty} \frac{1}{n^2} = 0 \qquad \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = e$$

Euler's number

Tip: implement sequences as a functional interface.

Recursive Sequences

✓ **Each term of sequence depends on other terms**

- Example:
  - **Fibonacci**: $a_n = a_{n-1} + a_{n-2}$
  - **Geometric** sequence: $a_n = r\, a_{n-1}$ → r is a constant number: $a_n = a_1 r^{n-1}$
  - **Factorial**: $a_n = n\, a_{n-1}$ → n is not a constant number
- Typically **initial conditions** are required. The number of initial conditions depends on the number of terms used in the recursion
  - Fibonacci: $a_1 = 1$, $a_2 = 1$, Factorial: $a_0 = 1$ or $a_1 = 1$ ($a_0 = a_1$ by definition)
  - Geometric: a_1 = some number
- Recursive sequences can have closed-form solutions
  - If possible implement the closed-form solutions
    - Geometric: a_n = some number * r^{n-1}
- If closed-form solution is hard to find, directly implement recursion in java

Series

✓ **A sum over the terms of a sequence**

variable

$$S_n = \sum_{i=k}^{n} a_i = \underbrace{a_k + a_{k+1} + \dots + a_n}_{\text{Number of terms} = \mathbf{n-k+1}}$$

Summation index

fixed

- Given a sequence we can sum over its terms
- Typically k = 0 or k = 1, n = 1, 2, 3, …
- In order to evaluate S_n we need to evaluate a_i terms
  - **Caching**
    - when evaluating S_n, save a_k, a_{k+1}, …, a_n
- Series: limit of S_n when n → ∞
  - Series converges if the limit exists. $|S_n - S_{n-1}| < \epsilon \Rightarrow \text{convergence}$
    - If a_n is monotonically increasing → series **always diverges**
  - Necessary condition for convergence: $a_n = S_n - S_{n-1}$ thus **lim a_n → 0**
- Use an Enum type for +infinity and –infinity
- Implement convergence test when one of the sum indices is infinity (error of the evaluation)

Tip: implement <u>caching</u> to avoid recalculation of the sequence.

Series: Test for convergence

✓ **What criterion should we use for convergence test?**

- We don't know the **exact** value of the limit
- Error must be defined without the exact value
- Absolute error
  - AbsError = |estimate(n) – exact value |
  - **Error sequence**: AbsError(n) = |estimate(n+1) – estimate(n)|
- Relative error
  - RelError = AbsError/|exact value|
    - What if |exact value| = 0?
  - RelError(n) = AbsError(n)/|estimate(n)|
- If a series converges
  - **AbsError → 0**, AbsError(n) → 0
  - **RelError → 0**, RelError(n) → 0

Speed of convergence

Series: Test for convergence

- Some tests that can determine the convergence of a series
- **Ratio** Test (D'Alembert's criterion)

$$\lim_{n\to\infty}\left|\frac{a_{n+1}}{a_n}\right| = r \begin{cases} r < 1\text{: convergent} \\ r > 1\text{: divergent} \\ r = 1\text{: inconclusive} \end{cases}$$

Partial sum

Series

$$s_n = \sum_{k=0}^{n} a_k \to s = \sum_{k=0}^{\infty} a_k = ?$$

- **Root** test

$$r = \lim_{n\to\infty}\sqrt[n]{|a_n|} \begin{cases} r < 1\text{: convergent} \\ r > 1\text{: divergent} \\ r = 1\text{: inconclusive} \end{cases}$$

Determines the radius of convergence of power series

- **Leibniz** criterion (Alternating series test, $a_n>0$, $(-1)^n$)

$$\boxed{\begin{gathered}\lim_{n\to\infty} a_n = 0 \\ a_{n+1} \le a_n \text{ for every } n\end{gathered}} \begin{cases} \sum_{n=k}^{\infty}(-1)^n a_n & \text{convergent} \\ \sum_{n=k}^{\infty}(-1)^{n+1} a_n & \text{convergent} \end{cases}$$

For a list of full tests, see:

https://en.wikipedia.org/wiki/Convergence_tests

Convergence Acceleration

Series acceleration

From Wikipedia, the free encyclopedia

In mathematics, series acceleration is one of a collection of sequence transformations for improving the rate of convergence of a series. Techniques for series acceleration are often applied in numerical analysis, where they are used to improve the speed of numerical integration. Series acceleration techniques may also be used, for example, to obtain a variety of identities on special functions. Thus, the Euler transform applied to the hypergeometric series gives some of the classic, well-known hypergeometric series identities.

- **What is the meaning of accelerated convergence?**
  - Consider sequence $\{a_n\}$ that converges to **A**
  - Assume that we can construct from a_n a new sequence $\{b_n\}$ that converges to **A** and has the property: $$\lim_{n\to\infty} \frac{b_n - A}{a_n - A} = 0$$

    Example: $\mathbf{b_n = 0.5\,(a_n + a_{n+1})}$
  - We say that b_n has a **faster** convergence than a_n

Sequence Transformations

- Some common transformations
  - **Binomial** transform

$$\{a_n\} \to \{s_n\} \Rightarrow s_n = \sum_{k=0}^{n} (-1)^k \binom{n}{k} a_k$$

  - **Euler** transform

$$\{(-1)^n a_n\} \to \{(-1)^n s_n\} \Rightarrow \sum_{n=0}^{\infty} (-1)^n a_n = \sum_{n=0}^{\infty} (-1)^n \frac{(\Delta^n a)_0}{2^{n+1}}$$

  - **Aitken** transform
    - Aitken's delta-squared process $X = \{x_n\} \to AX = \{(AX)_n\}$

$$(AX)_n = x_{n+2} - \frac{(\Delta x_{n+1})^2}{\Delta^2 x_n} = x_{n+2} - \frac{(x_{n+2} - x_{n+1})^2}{x_{n+2} - 2x_{n+1} + x_n}$$

  - **Shanks** transform

$$A = \sum_{m=0}^{\infty} a_m \Rightarrow A_n = \sum_{m=0}^{n} a_m \Rightarrow S(A_n) = \frac{A_{n+1}A_{n-1} - A_n^2}{A_{n+1} - 2A_n + A_{n-1}} = A_{n+1} - \frac{(A_{n+1} - A_n)^2}{A_{n+1} - 2A_n + A_{n-1}}$$

Difference (**forward**): $\Delta a_n = a_{n+1} - a_n$

Difference (**reverse**): $\bar{\Delta} a_n = a_n - a_{n-1}$

We can apply transformations multiple times

$a_n \to A_n$

↓

$S(A_n)$

↓

$S^2(A_n) = S(S(A_n))$

↓

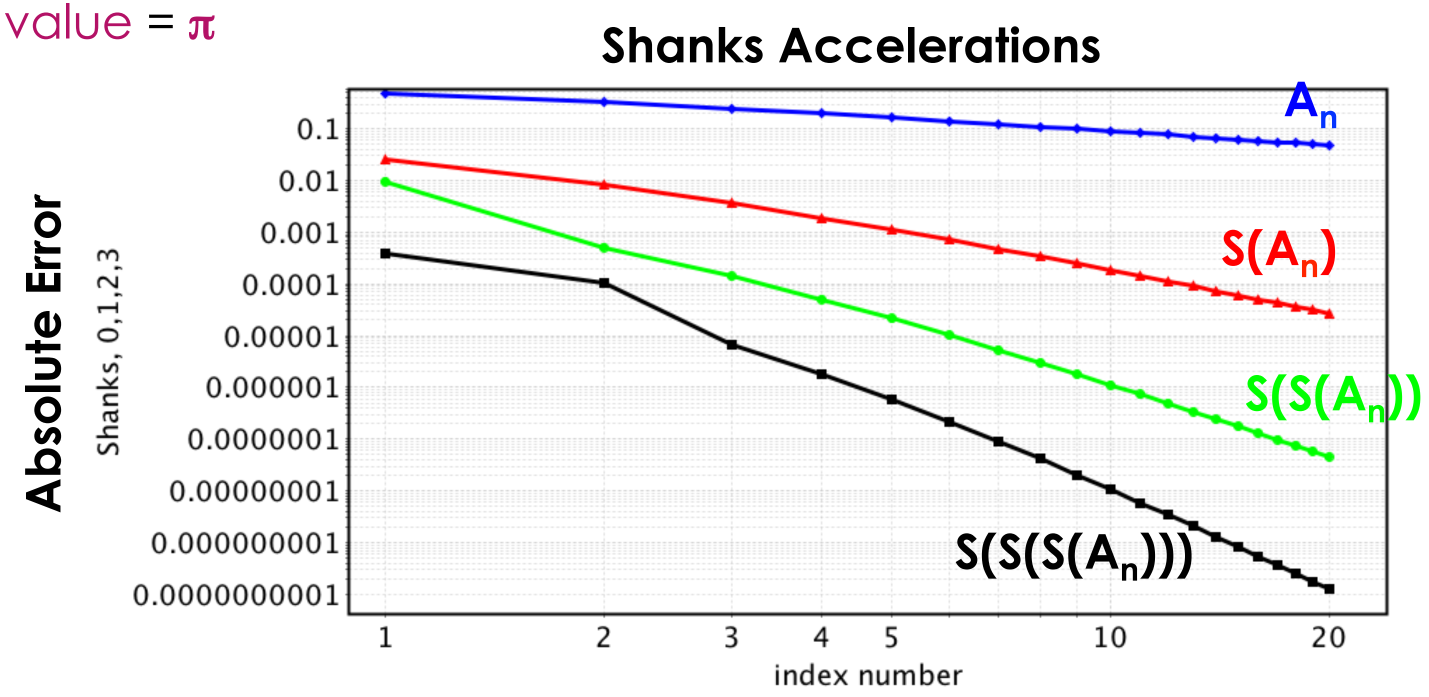
$S^3(A_n) = S(S(S(A_n)))$

Sequence Transformations

- **Java implementation**
- Approach 1: Add these transformations as **default** method to the Sequence interface
  - Keep the identity of this interface as a functional interface
- Approach 2: Create a new class that implements these transformations
- **Example**: use shanks transform to calculate the following series

$$\sum_{k=0}^{\infty} 4(-1)^k \frac{1}{2k+1} = 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots\right)$$ ➡ Exact value = π

n	A_n	$S(A_n)$	$S^2(A_n)$	$S^3(A_n)$
0	4.00000000	—	—	—
1	2.66666667	3.16666667	—	—
2	3.46666667	3.13333333	3.14210526	—
3	2.89523810	3.14523810	3.14145022	3.14159936
4	3.33968254	3.13968254	3.14164332	3.14159086
5	2.97604618	3.14271284	3.14157129	3.14159323
6	3.28373848	3.14088134	3.14160284	3.14159244
7	3.01707182	3.14207182	3.14158732	3.14159274
8	3.25236593	3.14125482	3.14159566	3.14159261
9	3.04183962	3.14183962	3.14159086	3.14159267
10	3.23231581	3.14140672	3.14159377	3.14159264
11	3.05840277	3.14173610	3.14159192	3.14159266
12	3.21840277	3.14147969	3.14159314	3.14159265



Richardson Transformation

- **Shanks & Aitken**
  - Good for alternating (oscillating) sequences → a_n has $(-1)^n$
  - They do not work well for non-oscillatory sequences
- **Richardson transform** is a better way for non-oscillatory sequences

Partial sum

$$\underbrace{s_n}_{\text{Partial sum}} = \sum_{k=0}^{n} a_k \to \underbrace{s}_{\text{Series}} = \sum_{k=0}^{\infty} a_k = ?$$

Series

$$R_1 = R(s_n) = (n+1)s_{n+1} - ns_n$$

```
/**
 * Richardson acceleration of a sequence.
 * @return returns a {@code Sequence} object
 */
default Sequence richardson() {
     return n -> (n+1)*evaluate(n+1)-n*evaluate(n) ;
}
```

$$R_2 = \frac{(n+2)^2 s_{n+2} - 2(n+1)^2 s_{n+1} + n^2 s_n}{2!}$$

$$R_3 = \frac{(n+3)^3 s_{n+3} - 3(n+2)^3 s_{n+2} + 3(n+1)^3 s_{n+1} - n^3 s_n}{3!}$$

$$R_4 = \frac{(n+4)^4 s_{n+4} - 4(n+3)^4 s_{n+3} + 6(n+2)^4 s_{n+2} - 4(n+1)^2 s_{n+1} + n^4 s_n}{4!}$$

Remember: Straight-up summing the terms of a sequence is the worst thing you can do !!!

Richardson Transformation

Partial sum

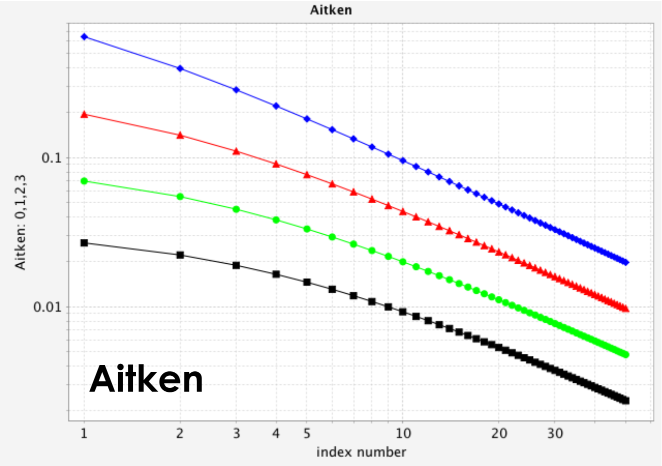
$$s_n = \sum_{k=0}^{n} a_k \to s = \sum_{k=0}^{\infty} a_k = ?$$

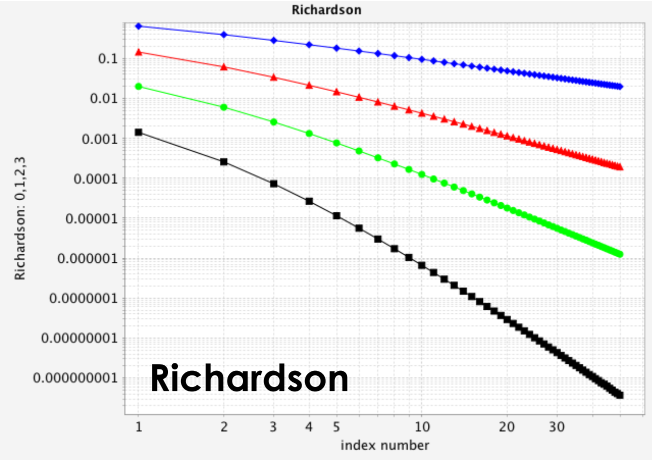
Series

- **Shanks & Aitken**
  - Good for alternating (oscillating) sequences → a_n has $(-1)^n$
  - They do not work well for non-oscillatory sequences
- **Richardson transform** is a better way for non-oscillatory sequences
- Example: $\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}$ ➡ Monotonic (non-oscillatory) convergence
- Let's compare
  - Aitken
  - Richardson
- **Conclusion**
  - Richardson is much better in this case!

Behavior of the error

$$s_n \sim L + \frac{A_1}{n} + \frac{A_2}{n^2} + \cdots$$





Tip: Understanding the behavior of Error is key to choosing the right accelerator.

Richardson Transformation

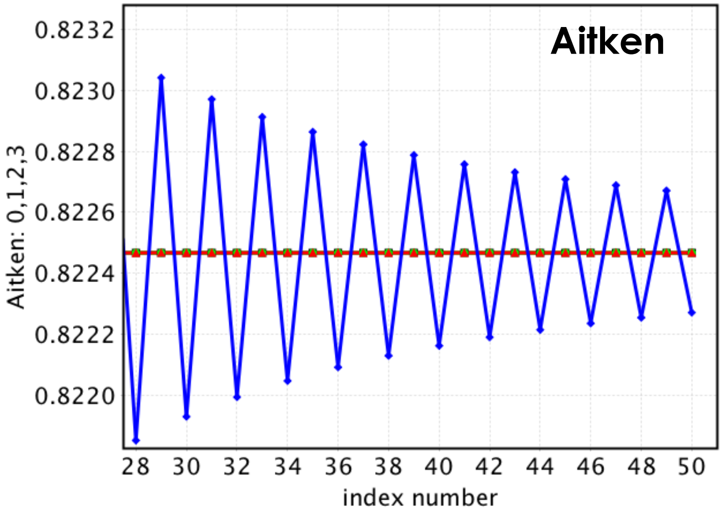
- **Shanks & Aitken**
  - Good for alternating (oscillating) sequences → a_n has $(-1)^n$
  - They do not work well for non-oscillatory sequences
- **Richardson transform** is a better way for non-oscillatory sequences

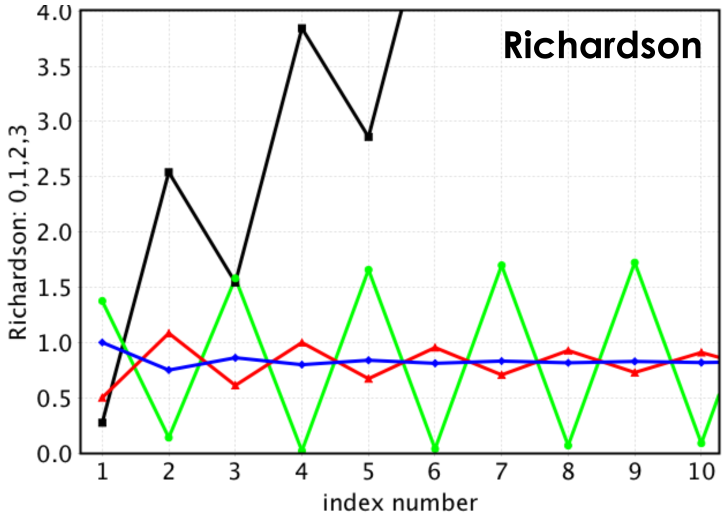
Partial sum

$$s_n = \sum_{k=0}^{n} a_k \to s = \sum_{k=0}^{\infty} a_k = ?$$

Series

- Example: $\displaystyle\sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k^2}$ ➡ Oscillatory convergence → Must converge
- Let's compare
  - Aitken
  - Richardson
- **Conclusion**
  - Richardson causes divergence!

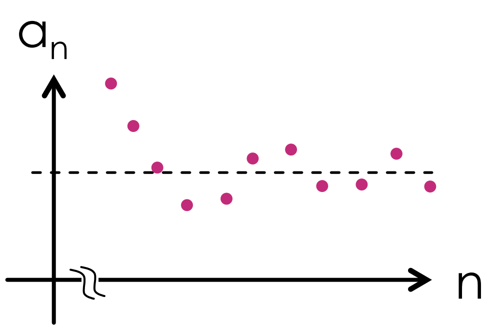




Tip: Understanding the behavior of Error is key to choosing the right accelerator.

Sequence Transformations

- Where do the ideas for Aitken, Shanks, and Richardson acceleration come from?
- Based on **modeling the asymptotic behavior of the error**
  - Alternating convergence → **Aitken** or **Shanks**
  - Uniform convergence → **Richardson**
- Eliminating poorly converging leading terms of the error



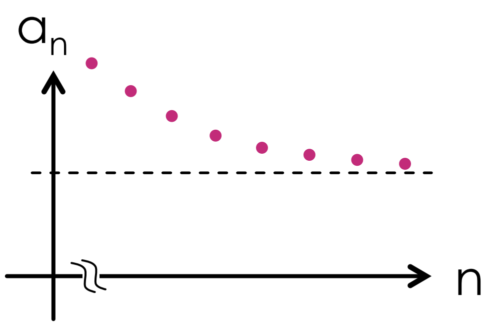
Oscillatory behavior

$$\boxed{a_n \sim L + A\,(-B)^n}$$

as n → ∞

|B|<1, A>0

Aitken Shanks



Non-oscillatory behavior

$$\boxed{a_n \sim L + \frac{A}{n^B}}$$ as n → ∞

B>=1 , A>0 → from above

B>=1 , A<0 → from below

→ **Richardson**

$$a_n \sim L + A\,(-B)^n + \cdots$$

$$a_{n+1} \sim L + A\,(-B)^{n+1}$$

$$a_{n-1} \sim L + A\,(-B)^{n-1}$$

Solve for L by eliminating A and B

Shanks $$L = \frac{a_n^2 - a_{n+1}\,a_{n-1}}{2a_n - a_{n+1} - a_{n-1}}$$

$$-B = \frac{a_n - L}{a_{n-1} - L} = \frac{a_{n+1} - L}{a_n - L}$$

Romberg Integration

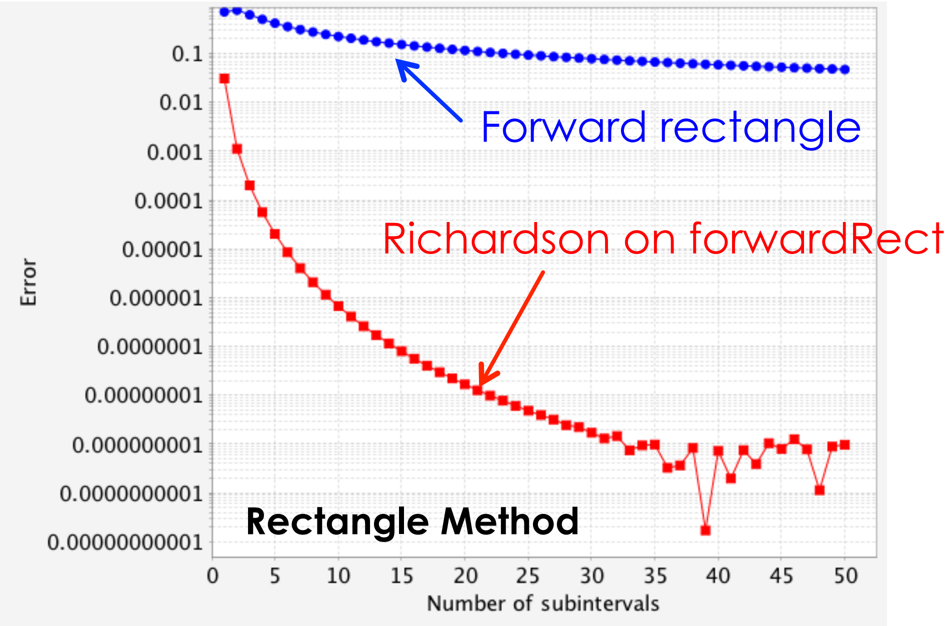
- Numerical integration follows Richardson error model → use Richardson acceleration
- Apply Richardson directly to the Trapezoid or Simpson methods

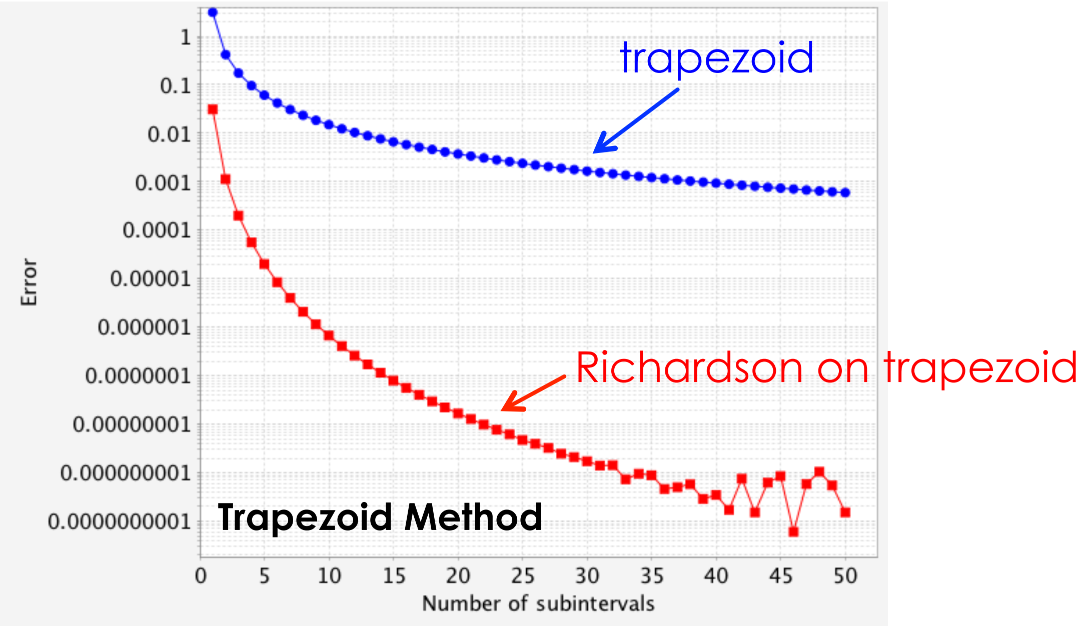
```
public double romberg(double start, double end, int subIntervals) {
    Sequence trapezoidSeq = n -> trapezoid(start, end, (int) n) ;
    Sequence rombergSeq = trapezoidSeq.richardson4() ;
    return rombergSeq.evaluate(subIntervals) ;
}
```

Applying 4th-order Richardson acceleration to the trapezoid summation

- **Comparison the converge**

f(x) = sin(x)





Useful Sequence Operations

- Useful relations for working with perturbation theory
  - Kronecker Delta sequence
    - δ[n] = 1 for n=0, δ[n] = 0 for n≠0
    - Add it as a static method to the Sequence interface
  - Unit Step sequence
    - It's a summation on the delta sequence
    - u[n] = 1 for n>=0, u[n] = 0 for n<0
  - Convolution Sum sequence
    - We assume that sequences are zero before n = 0 or n = 1

$$c_n = a_n \circledast b_n \Rightarrow c_n = \sum_{m=0\, or\, 1}^{n} a_m\, b_{n-m}$$

  - Running Average sequence

$$b_n = \frac{1}{M}\left(a_n + a_{n+1} + \cdots + a_{n+M-1}\right)$$

Operator overloading

- Addition
- Subtraction
- Multiplication
- Division

Default methods

Assumption: no negative index
a_n =0 for n<0
b_n = 0 for n<0

Perturbation Theory

✓ **Turning a hard problem into an infinite number of easy problems**

- Introduce a free parameter, typically called ε
- When setting ε=0, the solution is known → a_0
- Write a series in powers of ε ➡ $\sum_{n=0}^{\infty} a_n \epsilon^n$
  - Assuming analytic behavior
- Find a recursive relation for coefficients
  - Set ε = 1
  - Radius of convergence must be > 1
- Accelerate the convergence
  - If coefficients are oscillatory → Aitken or Shanks
  - If coefficients are monotonic → Richardson
- Exact = 0.754877666246693
- 7 term summation = **0.7543424**

$x^5 + x = 1$ ➡ Real root?

$\epsilon = 0$

Three cases

$\epsilon x^5 + x = 1$ ➡ x = 1

$x^5 + \epsilon x = 1$ ➡ x = 1 **Works**

$x^5 + x = \epsilon$ ➡ x = 0

$$x^5 + \epsilon x = 1$$

General form of the solution

$$x(\epsilon) = \sum_{n=0}^{\infty} a_n \epsilon^n$$

$$a_0 = 1,\ a_1 = \frac{-1}{5},\ a_2 = \frac{-1}{25}$$

Aitken or Shanks ⬅ $a_3 = \frac{-1}{125},\ a_4 = 0,\ a_5 = \frac{21}{15625},\ a_6 = \frac{78}{78125}$

Perturbation Theory

- Useful relations for working with perturbation theory
  - Adding a constant to a series

Kronecker delta

$$A + \sum_{n=0}^{\infty} a_n x^n = \sum_{n=0}^{\infty} b_n x^n \Rightarrow b_n = a_n + A\,\delta_{n,0}$$

  - Adding a power term to the series

$$A\,x^p + \sum_{n=0}^{\infty} a_n x^n = \sum_{n=0}^{\infty} b_n x^n \Rightarrow b_n = a_n + A\,\delta_{n,p}$$

  - Raising a series to some power

Convolution sum

$$\left(\sum_{n=0}^{\infty} a_n x^n\right)^M = \sum_{n=0}^{\infty} b_n x^n \Rightarrow b_n = a_n \circledast a_n \circledast a_n \circledast \cdots \circledast a_n$$

  - Multiplication

Convolution sum

$$\left(\sum_{n=0}^{\infty} a_n x^n\right)\left(\sum_{n=0}^{\infty} b_n x^n\right) = \sum_{n=0}^{\infty} c_n x^n \Rightarrow c_n = a_n \circledast b_n$$

Rate and Order of Convergence

- Consider the error model for Richardson Extrapolation
  - In this model for the error, we say that ***B*** is the order of convergence

$$a_n \sim L + \frac{A}{n^B} + \frac{A_1}{n^{B+1}} + \frac{A_2}{n^{B+2}} + \cdots$$

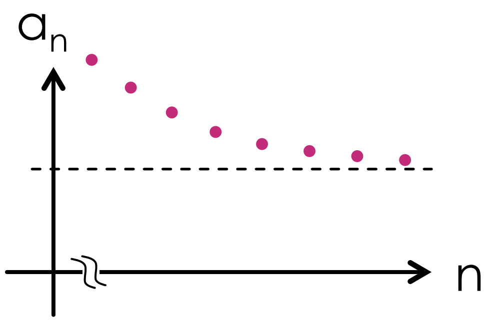
sequence

Asymptotic (as n→∞)

limit

Leading term of error

Order of convergence



Non-oscillatory behavior

$$\boxed{a_n \sim L + \frac{A}{n^B}}$$ as n → ∞

B>=1 , A>0 ➡ from above

B>=1 , A<0 ➡ from below

- Formal definition of the rate of convergence
  - Linear convergence $\lim_{k\to\infty} \frac{|a_{k+1} - L|}{|a_k - L|} = \mu$

    Rate of convergence $0 \le \mu \le 1$

  - Sublinear $\lim_{k\to\infty} \frac{|a_{k+1} - L|}{|a_k - L|} = 1$

    Special case

    Logarithmic $\lim_{k\to\infty} \frac{|a_{k+2} - a_{k+1}|}{|a_{k+1} - a_k|} = 1$

  - Superlinear $\lim_{k\to\infty} \frac{|a_{k+1} - L|}{|a_k - L|} = 0$

Numerical **differentiation** and **integration** have sublinear convergence

Rate and Order of Convergence

- Consider the error model for Richardson Extrapolation
  - Richardson-suited sequences typically have sublinear convergence

We don't like sublinear convergence… ☹☹☹

Sequence $a_n \sim L + \frac{A}{n^B} + \frac{A_1}{n^{B+1}} + \frac{A_2}{n^{B+2}} + \cdots \Rightarrow \lim_{k\to\infty} \frac{|a_{k+1} - L|}{|a_k - L|} = 1$

  - Aitken (Shanks)-suited sequences typically have linear convergence

Smaller is better!

Sequence $a_n \sim L + A(-B)^n + A_1(-B)^{n+1} + \cdots \Rightarrow \lim_{k\to\infty} \frac{|a_{k+1} - L|}{|a_k - L|} = |B| < 1$

  - This is the reason why Aitken (Shanks) does not work on Richardson-suited sequences
    - Sequence accelerators do not change the rate of convergence, they only change the order of convergence

$a_n \Rightarrow b_n = R_1(a_n) \Rightarrow b_n \sim L + \frac{C_1}{n^{B+1}} + \cdots \Rightarrow \lim_{k\to\infty} \frac{|b_{k+1} - L|}{|b_k - L|} = 1$

Richardson transformation does not change the rate of convergence

$a_n \Rightarrow b_n = S(a_n) \Rightarrow b_n \sim L + C_1(-B)^{n+1} + \cdots \Rightarrow \lim_{k\to\infty} \frac{|b_{k+1} - L|}{|b_k - L|} = |B| < 1$

Shanks transformation does not change the rate of convergence

Rate and Order of Convergence

- **General Definition**
  - Consider the following limit
    - $M \neq \infty$

$$\lim_{k\to\infty} \frac{|a_{k+1} - L|}{|a_k - L|^p} = \mu \leq M$$ → larger p and smaller μ indicate faster convergence

  - **p** is called the **order of convergence** ($p \geq 1$)
  - p =1 and μ = 1 → sublinear
  - p = 1 and 0<μ<1 → linear
  - p = 1 and μ = 0 → superlinear
  - p = 2 and μ > 0 → quadratic convergence
  - p = 3 and μ > 0 → cubic convergence

$$e_n = |x_n - L| = \frac{1}{2^n} \Rightarrow \frac{e_{n+1}}{e_n^p} = \begin{cases} 0 & p < 1 \\ 0.5 & p = 1 \\ \infty & p > 1 \end{cases}$$ sublinear

suitable for Aitken (Shanks)

$$e_n = |x_n - L| = \frac{1}{2^{2^n}} \Rightarrow \frac{e_{n+1}}{e_n^p} = \begin{cases} 0 & p < 2 \\ 1 & p = 2 \\ \infty & p > 2 \end{cases}$$ quadratic

- Most common
  - Linear and sublinear convergence
- **Newton-Raphson root finding** iterations
  - Quadratic convergence

Rate and Order of Convergence

- **How to estimate order of convergence?**
  - Consider the following limit
    - $M \neq \infty$

$$\lim_{k\to\infty} \frac{|a_{k+1} - L|}{|a_k - L|^p} = \mu \leq M$$

- A practical way to estimate p is:
  - For large k
  - In the limit k → ∞

$$p \approx \frac{\log\left(\frac{\Delta a_k}{\Delta a_{k-1}}\right)}{\log\left(\frac{\Delta a_{k-1}}{\Delta a_{k-2}}\right)}$$

Forward difference

- How to find rate of convergence (μ)?
  - We don't know the limit…
  - We may take use of the re-defined error
    - for superlinear sequences: $e_n \sim \tilde{e}_n$

Actual error sequence

Approximated error sequence

$$e_n = |a_n - L| \Rightarrow \tilde{e}_n = |a_{n+1} - a_n| \Rightarrow \begin{cases} \lim_{n\to\infty} \tilde{e}_n = 0 \\ \lim_{n\to\infty} e_n = 0 \end{cases}$$