ex_01

Nikolai German

2025-04-23

Aufgabe 1

b) Euclid's greatest common denominator

```
euclid <- function(m, n) {
  checkmate::assertIntegerish(m, lower = 0, any.missing = FALSE, len = 1)
  checkmate::assertIntegerish(n, lower = 0, any.missing = FALSE, len = 1)
  if (n == 0) return(m)
  if (m == 0) return(n)
  cat(n, m, "\n")
  if (m <= n) {
    return(euclid(m, n %% m))
  } else {
    return(euclid(m %% n, n))
  }
}

euclid(270, 192)
```

```
## 192 270
## 192 78
## 36 78
## 36 6
```

```
## [1] 6
```

Aufgabe 3

Numeric Differentation

```
derive <- function(f, x0, h) {
  checkmate::assertFunction(f, args = c("x"))
  checkmate::assertNumeric(x0, any.missing = FALSE, len = 1)
  #checkmate::assertNumeric(h, any.missing = FALSE, len = 1, lower = 0)
  (f(x0 + h) - f(x0)) / h
}
```

initialize functions and their analytical derivatives:

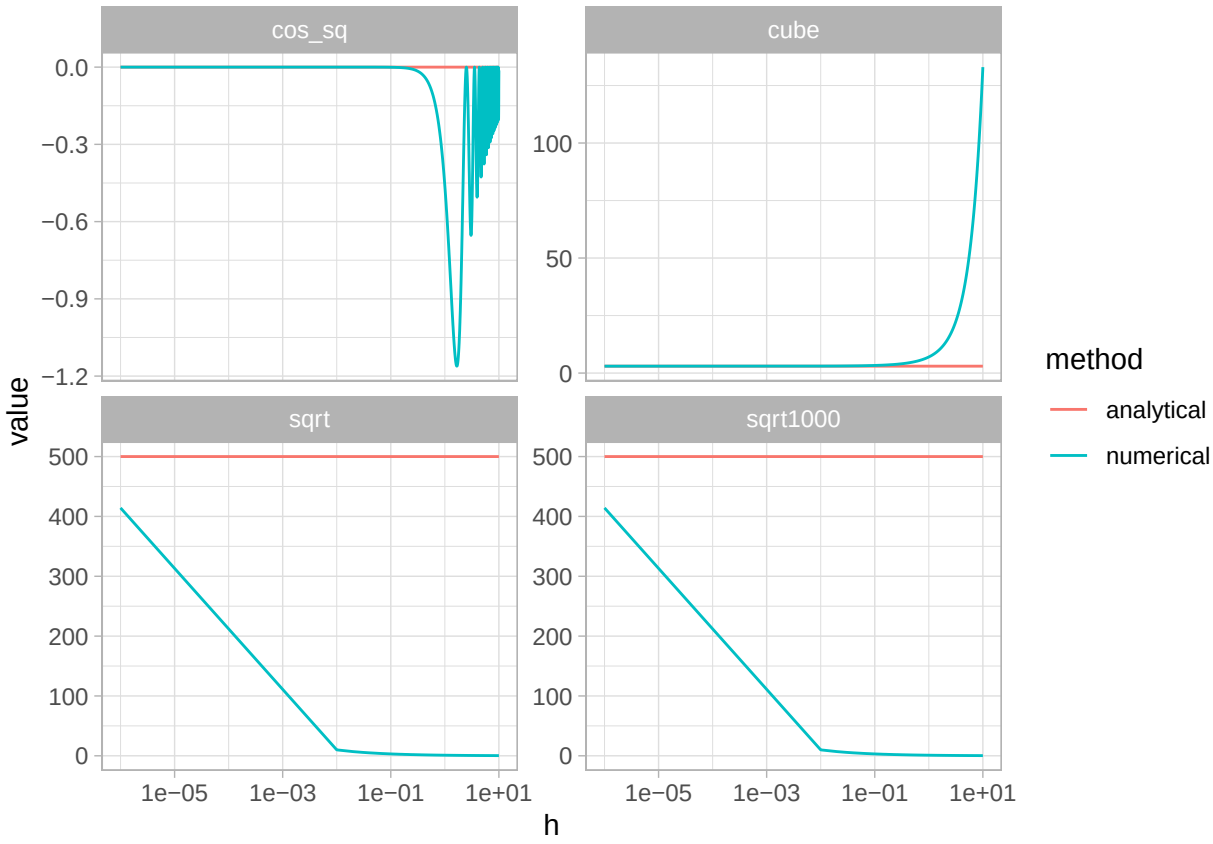
```
cube <- function(x) x^3
cube_ana <- function(x) 3*x^2

cos_sq <- function(x) cos(x^2)
```

```
cos_sq_ana <- function(x) -2*x*sin(x^2)

sqrt_ana <- function(x) 1 / (2*sqrt(x))

sqrt1000 <- function(x) sqrt(x - 1000)
sqrt1000_ana <- function(x) 1 / (2*sqrt(x - 1000))
```

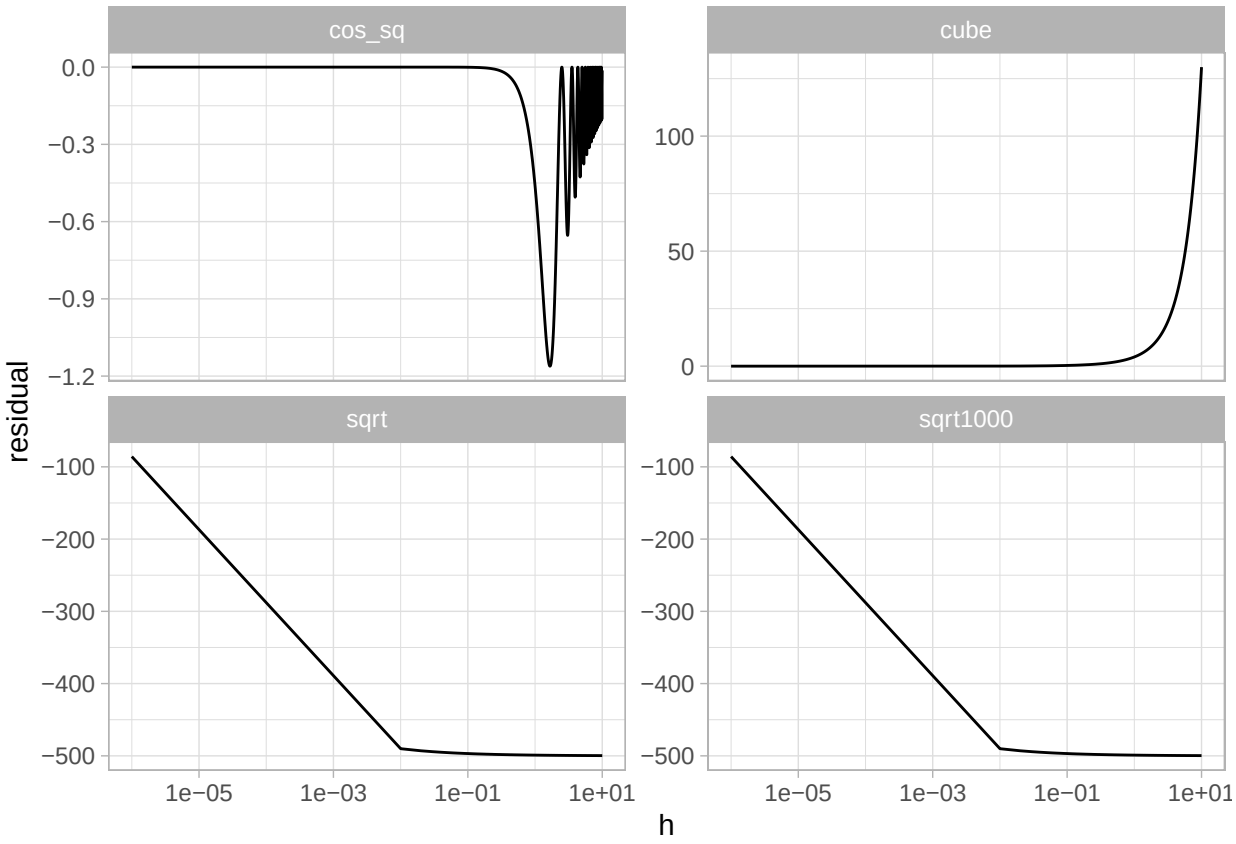
build some data:

```
truth <- tibble(cube = cube_ana(1),
           cos_sq = cos_sq_ana(1e-3),
           sqrt = sqrt_ana(1e-6),
           sqrt1000 = sqrt1000_ana(1000 + 1e-6)
) %>%
  pivot_longer(everything(), names_to = "fun", values_to = "analytical")

df <- tibble(h = seq(1e-6, 10, length.out = 1000)) %>%
  mutate(
    cube = derive(cube, 1, h),
    cos_sq = derive(cos_sq, 1e-3, h),
    sqrt = derive(sqrt, 1e-6, h),
    sqrt1000 = derive(sqrt1000, 1000 + 1e-6, h)
  ) %>%
  pivot_longer(-h, values_to = "numerical", names_to = "fun") %>%
  left_join(truth, by = "fun") %>%
  mutate(residual = numerical - analytical) %>%
  pivot_longer(c(numerical, analytical), names_to = "method", values_to = "value")
```

Let's look at the numerical vs. the analytical values:



And at the resiudals:



Exercise 4

a) Matrix Product

```
mat_prod <- function(a, b) {
  checkmate::assertMatrix(a, min.rows = 1, min.cols = 1, any.missing = FALSE)
  checkmate::assertMatrix(b, nrows = ncol(a), min.cols = 1, any.missing = FALSE)

  res <- matrix(nrow = nrow(a), ncol = ncol(b))

  for (i in seq_len(nrow(res))) {
    for (j in seq_len(ncol(res))) {
      res[i,j] <- sum(a[i, ] * b[, j])
    }
  }
  res
}
```

```
a <- matrix(runif(20), nrow = 4)
b <- matrix(rnorm(15), nrow = 5)

mat_prod(a, b)
```

```
##            [,1]       [,2]        [,3]
## [1,]  0.4560852 -0.3314048 -0.57756245
## [2,] -0.9370929  1.0015270  0.66416567
```

```
## [3,] -0.6646552  0.1974288 -0.02459352
## [4,]  0.4258936  0.5231516 -0.11107435
```

```
a %*% b
```

```
##            [,1]       [,2]        [,3]
## [1,]  0.4560852 -0.3314048 -0.57756245
## [2,] -0.9370929  1.0015270  0.66416567
## [3,] -0.6646552  0.1974288 -0.02459352
## [4,]  0.4258936  0.5231516 -0.11107435
```