apropl.r

denis

2021-07-16

#!/usr/bin/r  
  
# For different types of functions, different basis sets may be appropriate.  
set.seed(123)  
A1ps <- c(mp = 12.5, mc = 12.3, md = 12.4)  
A1ps

## mp mc md   
## 12.5 12.3 12.4

A2ps <- c(pm = 12.5, pm = 12.3, pm = 12.4)  
A2ps

## pm pm pm   
## 12.5 12.3 12.4

# ordered different value compared fireworks update laptop  
diff(A1ps >= A2ps)

## mc md   
## 0 0

# Polynomials are often used, and there are some standard sets of orthogonal   
# polynomials, such as Jacobi, Hermite, and so on.  
poly(A1ps, A2ps, degree = 1, coefs = NULL, raw = FALSE, simple = FALSE)

## 1.0 0.1  
## mp 7.071068e-01 7.071068e-01  
## mc -7.071068e-01 -7.071068e-01  
## md 9.813078e-17 9.813078e-17  
## attr(,"degree")  
## [1] 1 1  
## attr(,"coefs")  
## attr(,"coefs")[[1]]  
## attr(,"coefs")[[1]]$alpha  
## [1] 12.4  
##   
## attr(,"coefs")[[1]]$norm2  
## [1] 1.00 3.00 0.02  
##   
##   
## attr(,"coefs")[[2]]  
## attr(,"coefs")[[2]]$alpha  
## [1] 12.4  
##   
## attr(,"coefs")[[2]]$norm2  
## [1] 1.00 3.00 0.02  
##   
##   
## attr(,"class")  
## [1] "poly" "matrix"

# For periodic functions especially, orthogonal trigonometric   
# functions are useful.  
trigamma(A1ps)

## mp mc md   
## 0.08328522 0.08469517 0.08398428

trigamma(A2ps)

## pm pm pm   
## 0.08328522 0.08469517 0.08398428

# 2.2.6 Approximation of Vectors In high-dimensional vector spaces, it is often   
# useful to approximate a given vector in terms of vectors from a lower   
# dimensional space.  
cov(A1ps, A2ps)

## [1] 0.01

# Suppose, for example, that V ⊂ IR n is a vector space of dimension k   
# (necessarily, k ≤ n) and x is a given n-vector.  
k = 5  
n = 1  
  
if (k < n) {  
 print(k < n)  
} else {  
 vector(mode = "logical", length = 0L)  
}

## logical(0)

# pipelines type procedure pass checkup method  
# buffer lines ...  
# We wish to determine a vector x̃ in V that approximates x.  
Vx <- c(c(runif(n, min = 0, max = 1), open = 1,   
 encoding = getOption("A2ps")))  
Vx

## open   
## 0.2875775 1.0000000

# Optimally of the Fourier Coefficients  
googledrive::as\_id("http://a2ps.vx")

## [1] NA  
## attr(,"class")  
## [1] "drive\_id"

attr(A2ps, which = "http://a2ps.vx", exact = "http://a2ps.vx")

## NULL

drop(n + 1)

## [1] 2

window(k \* n)

## [1] 5  
## attr(,"tsp")  
## [1] 1 1 1

attr(A2ps, which = "http://a2ps.vx", exact = "http://a2ps.vx/tse")

## NULL

# One obvious criterion would be based on a norm of the difference of the given   
# vector and the approximating vector.  
outer(A2ps, A1ps, FUN = "\*")

## mp mc md  
## pm 156.25 153.75 155.00  
## pm 153.75 151.29 152.52  
## pm 155.00 152.52 153.76

# lollipop val breadcrumb water  
loll <- c(pop = 2.4, bread = 2.4, water = 2.4)  
loll

## pop bread water   
## 2.4 2.4 2.4

# This difference is a truncation error. Let u 1 , . . . , u k be an   
# orthonormal basis set for V, and let  
u1 <- trunc(loll, A1ps)  
u1

## pop bread water   
## 2 2 2

# where the c i are the Fourier coeﬃcients of x, x, u i . Now l  
# et v = a 1 u 1 + · · · + a k u k be any other vector in V, and consider  
xx <- c(from = u1, to = loll, strict = TRUE)  
xx

## from.pop from.bread from.water to.pop to.bread to.water strict   
## 2.0 2.0 2.0 2.4 2.4 2.4 1.0

# Therefore we have x − x ≤ x − v, and so x̃ is the best approximation of  
# x with respect to the Euclidean norm in the k-dimensional vector space V.  
if (xx != xx || xx != xx){  
 c(xx) + spacetime::geometry(obj = u1)  
}  
xx

## from.pop from.bread from.water to.pop to.bread to.water strict   
## 2.0 2.0 2.0 2.4 2.4 2.4 1.0

# Choice of the Best Basis Subset  
# Now, posing the problem another way, we may seek the best k-dimensional  
# subspace of IR n from which to choose an approximating vector.  
IR <- subset.default(xx, missing(u1), A2ps)  
IR

## named numeric(0)

# This question  
# is not well-posed (because the one-dimensional vector space determined by x  
# is the solution), but we can pose a related interesting question: suppose we  
# have a Fourier expansion of x in terms of a set of n orthogonal basis vectors,  
# u 1 , . . . , u n , and we want to choose the “best” k basis vectors from this   
# set and use them to form an approximation of x.  
k = n  
un <- c(k, color = 99, conf.level = 0.95,   
 std = 12, margin = c(1, 2),  
 space = 0.2, main = NULL, mfrow = NULL, mfcol = NULL)  
un

## color conf.level std margin1 margin2 space   
## 1.00 99.00 0.95 12.00 1.00 2.00 0.20

# (This restriction of the problem is  
# equivalent to choosing a coordinate system.)  
later::create\_loop(autorun = FALSE)

## <event loop>  
## id: 1

# We see the solution immediately  
# from inequality (2.42): we choose the k u i s corresponding to the k   
# largest c i s  
# in absolute value, and we take  
call("k", u1, c(i = 2.42))

## k(c(pop = 2, bread = 2, water = 2), c(i = 2.42))

call("n", xx)

## n(c(from.pop = 2, from.bread = 2, from.water = 2, to.pop = 2.4,   
## to.bread = 2.4, to.water = 2.4, strict = 1))

outer(n, xx, FUN = "\*")

## from.pop from.bread from.water to.pop to.bread to.water strict  
## [1,] 2 2 2 2.4 2.4 2.4 1

# 2.2.7 Flats, Acne Spaces, and Hyperplanes  
# Given an n-dimensional vector space of order n, IR n for example, consider a  
# system of m linear equations in the n-vector variable x,  
Acne <- c(n, k, A2ps)  
Acne

## pm pm pm   
## 1.0 1.0 12.5 12.3 12.4

m = 12  
n = 1  
  
# where c 1 , . . . , c m are linearly independent n-vectors (and hence m ≤ n).  
c1 <- m < n  
c1

## [1] FALSE

# The set of points defined by these linear equations is called a rat.  
PlantGrowth

## weight group  
## 1 4.17 ctrl  
## 2 5.58 ctrl  
## 3 5.18 ctrl  
## 4 6.11 ctrl  
## 5 4.50 ctrl  
## 6 4.61 ctrl  
## 7 5.17 ctrl  
## 8 4.53 ctrl  
## 9 5.33 ctrl  
## 10 5.14 ctrl  
## 11 4.81 trt1  
## 12 4.17 trt1  
## 13 4.41 trt1  
## 14 3.59 trt1  
## 15 5.87 trt1  
## 16 3.83 trt1  
## 17 6.03 trt1  
## 18 4.89 trt1  
## 19 4.32 trt1  
## 20 4.69 trt1  
## 21 6.31 trt2  
## 22 5.12 trt2  
## 23 5.54 trt2  
## 24 5.50 trt2  
## 25 5.37 trt2  
## 26 5.29 trt2  
## 27 4.92 trt2  
## 28 6.15 trt2  
## 29 5.80 trt2  
## 30 5.26 trt2