Package 'MCMChybridGP'

November 12, 2020

Version 5.4

Title Hybrid Markov Chain Monte Carlo using Gaussian Processes

Author Mark J. Fielding <mark.fielding@gmx.com>

Maintainer Mark J. Fielding <mark.fielding@gmx.com>

Depends MASS

Description Hybrid Markov chain Monte Carlo (MCMC) to simulate from a multimodal target distribution. A Gaussian process approximation makes this possible when derivatives are unknown. The Package serves to minimize the number of function evaluations in Bayesian calibration of computer models using parallel tempering. It allows replacement of the true target distribution in high temperature chains, or complete replacement of the target. Methods used are described in, ``Efficient MCMC schemes for computationally expensive posterior distributions", Fielding et al. (2011) <doi:10.1198/TECH.2010.09195>. The research presented in this work was carried out as part of the Singapore-Delft Water Alliance Multi-Objective Multi-Reservoir Management research programme (R-264-001-272).

License GPL-2

LazyLoad yes

Repository CRAN

Date/Publication 2020-11-12 18:50:02 UTC

NeedsCompilation yes

R topics documented:

MCMChybridGP-package *Hybrid MCMC for a multimodal density with derivatives replaced by Gaussian process*

Description

Hybrid Markov chain Monte Carlo (MCMC) to simulate from a multimodal target distribution with derivatives unknown. A Gaussian process fit is used to approximate derivatives. The Package consists of an Exploratory phase, with `hybrid.explore`, followed by a Sampling phase, with `hybrid.sample`. The user is to supply the log-density `f` of the target distribution along with a small number of (say 10) points to get things started. The Sampling phase allows replacement of the true target in high temperature chains, or complete replacement of the target. A full description of the method is given in Fielding, Nott and Liong (2011).

The authors gratefully acknowledge the support & contributions of the Singapore-Delft Water Alliance. The research presented in this work was carried out as part of the Multi-Objective Multi-Reservoir Management research programme (R-264-001-272).

Details

Package:	MCMChybridGP
Type:	Package
Version:	1.0
Date:	2009-09-15
License:	GPL-2
LazyLoad:	yes

Author(s)

Mark James Fielding <mark.fielding@gmx.com>

Maintainer: Mark James Fielding <mark.fielding@gmx.com>

References

"Efficient MCMC Schemes for Computationally Expensive Posterior Distributions", Fielding, Nott and Liong (2011).

See Also

`hybrid.explore`, `hybrid.sample`

Examples

```
mu1 <- c(-1, -1)
```

```
mu2 <- c(+1, +1)
sigma.sq <- 0.1225
ub <- c(1.5, 3)
X0 <- generateX0(lb=c(-2,-2), ub=ub)
f <- function(x) {
    px <- 1/4/pi/sqrt(sigma.sq) * exp(-1/2/sigma.sq *
        sum((x - mu1)^2)) + 1/4/pi/sqrt(sigma.sq) *
        exp(-1/2/sigma.sq * sum((x - mu2)^2))
    return(log(px))
}


explore.out <- hybrid.explore(f, X0, ub=ub, n=150, graph=TRUE)
sample.out <- hybrid.sample(explore.out, n=500, graph=TRUE)

opar <- par(mfrow=c(2,1))
plot(density(sample.out$SAMP[,1]), xlab="x1", ylab="f(x)")
plot(density(sample.out$SAMP[,2]), xlab="x2", ylab="f(x)")
par(opar)
```

`generateX0` *Generate some initial points for the hybrid explore phase*

Description

A function to randomly generate `n0`-many points within requested bounds, `lb`, `ub`. The points selected are to have the largest, minimum distance between any two points.

Usage

```
generateX0 (lb, ub, n0 = 10, npool = 100)
```

Arguments

`lb`	Lower (finite) bounds for points generated.
`ub`	Upper (finite) bounds for points generated.
`n0`	The number of points to be generated.
`npool`	The size of the pool of sets of randomly generated points in determining the best set of points.

Value

A matrix is returned which can then be supplied to `hybrid.explore`.

Author(s)

Mark James Fielding <mark.fielding@gmx.com>

References

"Efficient MCMC Schemes for Computationally Expensive Posterior Distributions", Fielding, Nott and Liong (2011).

See Also

hybrid.explore,

Examples

```
lb <- c(-3,-3)
ub <- c(3,3)
X0 <- generateX0(lb, ub)
```

GProcess *Determine a Gaussian process fit to a multivariate log-density function.*

Description

A function to determine a Gaussian process fit to a set of points forming a matrix X, given a column of corresponding values of the log-density of a target distribution. Returned is a (zero mean) approximation Ef of the log-density and various components of the Gaussian process fit as used by hybrid.explore and hybrid.sample.

Usage

```
GProcess(X, y, params = NULL, request.functions = TRUE, finetune = FALSE)
```

Arguments

X	A matrix of at least 2 columns with rows representing the points (nodes) for a Gaussian process fit.
y	A column of corresponding values of the log-density. Each entry corresponds to the log-density evaluated at the respective row in X.
params	Gaussian process parameters as used in hybrid.explore. (Optimal values are determined (starting from a supplied value).)
request.functions	Optional boolean argument (default TRUE) to request the return of function approximations Ef (and sigmaf) of the log-density.
finetune	Optional boolean argument (default FALSE) to determine fine-tuned optimal values in params.

Value

Returned is a list as requested consisting of:

`Ef`	The Gaussian process approximation of the log-density function.
`sigmaf`	Upon request, a function giving the Gaussian process approximation of the standard deviation with respect to `y=f(x)` for a given point.
`Sigma`	Covariance matrix used in the Gaussian process fit.
`Sigma.inv`	The inverse of the Covariance matrix.
`inverseOK`	Boolean flag to indicate successful calculation of `Sigma.inv`.
`X`	The original `X` matrix as supplied.
`y`	`y` as supplied. (Note calculations use `y-mean(y)`)
`params`	Parameter values for the Gaussian process fit.

Author(s)

Mark James Fielding <mark.fielding@gmx.com>

References

"Efficient MCMC Schemes for Computationally Expensive Posterior Distributions", Fielding, Nott and Liong (2011).

See Also

`hybrid.explore`,

Examples

```
mu1 <- c(-1, -1)
mu2 <- c(+1, +1)
sigma.sq <- 0.1225
X <- matrix(c(-2,-1,0,-2,0,2,0,1,2, -2,-1,-2,0,0,0,2,1,2), ncol = 2)
f <- function(x) {
    px <- 1/4/pi/sqrt(sigma.sq) * exp(-1/2/sigma.sq *
        sum((x - mu1)^2)) + 1/4/pi/sqrt(sigma.sq) *
        exp(-1/2/sigma.sq * sum((x - mu2)^2))
    return(log(px))
}

y <- rep(NA, 9)
for(i in 1:9) y[i] <- f(X[i,])

Ef <- GProcess(X, y, request.functions = TRUE)$Ef
Ey <- NA*y
for(i in 1:9) Ey[i] <- Ef(X[i,])
data.frame(X, y, Ey)
## Gaussian process close to exact at points supplied.
```

`hybrid.explore` *Exploratory phase to determine points used for a Gaussian process fit.*

Description

The Exploratory phase of hybrid MCMC using a Gaussian process approximation of derivatives. The user must provide the log-density of a target distribution and a small set of (say 10) points forming the matrix X0. Using a Gaussian process approximation, points are added until a appropriate set of points are determined for a final Gaussian process fit to the target distribution. Results can then be passed to the Sampling phase or the Gaussian process approximation `Ef` can be used to approximate `f`.

Usage

```
hybrid.explore(f, X0, ..., y0 = NULL, n = 200, L = 1000, delta = 0.003,
                 nchains = 5, T.mult = 1.5, lb = NULL, ub = NULL,
                 maxleap=0.95, finetune = FALSE, Tinclude = nchains,
                 nswaps = choose(nchains, 2), graph = FALSE)
```

Arguments

`f`	A function giving the log-density of the target distribution.
`X0`	A matrix with rows giving say 10 points for an initial Gaussian process fit.
`...`	Further arguments to be passed to `f`.
`y0`	Corresponding function values if already known. Otherwise the corresponding function values will first be evaluated.
`n`	An optional integer argument. The number of points to be generated for the final Gaussian process fit.
`L`	An optional integer argument. The number of steps to be used in Leapfrog moves in MCMC proposals.
`delta`	An optional numeric argument (default 0.003). The size of Leapfrog steps to be made, with a suitable balance between `L` and `delta`.
`nchains`	An optional integer argument. The number of Parallel Tempering chains to be used (default 5).
`T.mult`	An optional numeric argument. The Temperature multiple to be used in Parallel Tempering (default 1.5).
`lb`	An optional numeric argument. Lower bounds placed on X. Only points within bounds are included in Gaussian process fit.
`ub`	An optional numeric argument. Upper bounds placed on X. Only points within bounds are included in Gaussian process fit.
`maxleap`	An optional numeric 0-1 argument (default 0.95). Used in early stops in Leapfrog moves. The Leapfrog is stopped when the standard deviation of `f(x)` exceeds an appropriate corresponding value.

finetune	An optional boolean argument (default FALSE). Option for fine-tuning Gaussian process parameters. This may be useful when a good fit is difficult to achieve.
Tinclude	An optional integer argument (default nchains). Limit the number of temperatures to contribute points to the Gaussian process fit. With low acceptance rates for T=1, Tinclude=1 might be used. Otherwise a larger value gives a faster explore phase.
nswaps	An optional integer argument. The number of repetitions of proposed swaps between the parallel chains.
graph	An optional boolean argument (default FALSE). To request a graphical display of progress during the explore phase.

Details

The method used in hybrid.explore is described in Fielding, Nott and Liong (2011).

Value

A list is returned to be used as input to hybrid.sample.

X	A matrix made up of the set of points used in the final Gaussian process fit.
y	A column of the corresponding evaluations of f.
f	The original log-density function supplied.
maxleap	The value of maxleap as default or supplied.
function.calls	The number of function calls used to evaluate f.
details	A list containing information particular to hybrid.sample.
GPfit	Ouput from GProcess with the final Gaussian process fit and in particular the (zero mean) approximation Ef.

Note

The methods used in hybrid.explore and hybrid.sample give extensions to the work of Rasmussen (2003), as described in Fielding, Nott and Liong (2011).

For very low acceptance rates, points included at later stages are likely to be more useful with a fit only deteriorated by the earlier points. In such a case a second run of hybrid.explore might be useful, taking values for X0 and y0 as those output for X and y from the first run.

Author(s)

Mark J. Fielding <mark.fielding@gmx.com>

References

"Efficient MCMC Schemes for Computationally Expensive Posterior Distributions", Fielding, Nott and Liong (2011).

See Also

hybrid.sample, GProcess, generateX0.

Examples

```
mu1 <- c(-1, -1)
mu2 <- c(+1, +1)
sigma.sq <- 0.16
ub <- c(1.5, 3)
X0 <- matrix(c(-2,-1, 0,-2, 0, 1, 0, 1, 1,
               -2,-1,-2, 0, 0, 0, 2, 1, 2), ncol = 2)
f <- function(x) {
    px <- 1/4/pi/sqrt(sigma.sq) * exp(-1/2/sigma.sq *
        sum((x - mu1)^2)) + 1/4/pi/sqrt(sigma.sq) *
        exp(-1/2/sigma.sq * sum((x - mu2)^2))
    return(log(px))
}


explore.out <- hybrid.explore(f, X0, ub=ub, n=150, graph=TRUE)
sample.out <- hybrid.sample(explore.out, n=500, graph=TRUE)

opar <- par(mfrow=c(2,1))
plot(density(sample.out$SAMP[,1]), xlab="x1", ylab="f(x)")
plot(density(sample.out$SAMP[,2]), xlab="x2", ylab="f(x)")
par(opar)
```

---

hybrid.sample *Sampling phase applying results from Exploratory phase.*

Description

Sampling phase in hybrid MCMC, which takes the output from hybrid.explore to samples from the target distribution supplied. The number of chains, Leapfrog moves and Gaussian process parameters are the same as used in hybrid.explore, or the values may be updated here. For a target distribution time consuming to evaluate, the target can be replaced completely by the Gaussian process approximation in some or all of the chains. Bounds supplied act as reflecting barriers.

Usage

```
hybrid.sample(Explore, n = 1000, replace.target = c(0, 1, 2),
              lb = NULL, ub = NULL, L = NULL, delta = NULL,
              nchains = NULL, T.mult = NULL, maxleap = NULL,
              r = 5, nswaps = NULL, graph = FALSE)
```

Arguments

Explore	Output from hybrid.explore. A list object consisting of X, y, f and GPfit. X is the set of points from which a Gaussian process fit is gained. Then y is the corresponding values of the target log-density function f, with further arguments ... as supplied to hybrid.explore. If GPfit is NULL then a new Gaussian process fit is determined.
n	The number of sampling iterations.
L	An optional integer argument passed from hybrid.explore. The number of steps used in Leapfrog moves.
delta	An optional numerical argument passed from hybrid.explore. The size of steps used in Leapfrog moves.
lb	An optional numeric argument passed from hybrid.explore. Lower (finite) bounds placed on X. (If supplied, then ub must also be supplied and finite.)
ub	An optional numeric argument passed from hybrid.explore. Upper (finite) bounds placed on X. (If supplied, then lb must also be supplied and finite.)
nchains	An optional integer argument passed from hybrid.explore. The number of MCMC parallel chains to be used.
T.mult	An optional integer argument passed from hybrid.explore. The number of Parallel Tempering chains to be used.
maxleap	An optional numerical argument passed from hybrid.explore. Gives the maximum standard deviation of f(x) that a point can vary from points in Gaussian process fit.
r	An optional numerical argument (default 5). A penalty factor on points straying from the region of Gaussian process fit, when the target distribution is replaced.
nswaps	An optional integer argument passed from hybrid.explore. The number of repititions of swaps proposed between MCMC chains.
replace.target	The sampling scheme to be used (0, 1 or 2) in acceptance of MCMC proposals. Where 0 represents using the true target distribution in all chains. 1 (default) represents using the true target distribution only in the primary chain (having temperature 1). 2 represents replacing the target distribution in all chains by the Gaussian process approximation.
graph	An optional boolean argument (default is FALSE). Request graphical progress display during the sample phase.

Details

The method used in hybrid.sample is described in Fielding, Nott and Liong (2011).

Value

A list is returned consisting of the following.

SAMP	A matrix with rows corresponding to sampled points generated from the target distribution.

y	A column of the corrresponding values of the log-density of the target distribution.
acceptance	A column of 0 (rejected) and 1 (accepted) giving a record of sampling proposal acceptance.
function.calls	The number of function calls to evaluate the true log-density.

Note

A record is kept throughout a run of hybrid.sample stored as a global variable list, hybrid.sample.out. Useful for a run stopped prematurely.

The method used in hybrid.sample gives extensions to the work of Rasmussen (2003) and is described in Fielding, Nott and Liong (2011).

Author(s)

Mark J. Fielding <mark.fielding@gmx.com>

References

"Efficient MCMC Schemes for Computationally Expensive Posterior Distributions", Fielding, Nott and Liong (2011).

See Also

hybrid.explore

Examples

```
mu1 <- c(-1, -1)
mu2 <- c(+1, +1)
sigma.sq <- 0.16
ub <- c(1.5, 3)
X0 <- matrix(c(-2,-1, 0,-2, 0, 1, 0, 1, 1,
               -2,-1,-2, 0, 0, 0, 2, 1, 2), ncol = 2)
f <- function(x) {
    px <- 1/4/pi/sqrt(sigma.sq) * exp(-1/2/sigma.sq *
        sum((x - mu1)^2)) + 1/4/pi/sqrt(sigma.sq) *
        exp(-1/2/sigma.sq * sum((x - mu2)^2))
    return(log(px))
}


explore.out <- hybrid.explore(f, X0, ub=ub, n=150, graph=TRUE)
sample.out <- hybrid.sample(explore.out, n=500, graph=TRUE)

opar <- par(mfrow=c(2,1))
plot(density(sample.out$SAMP[,1]), xlab="x1", ylab="f(x)")
plot(density(sample.out$SAMP[,2]), xlab="x2", ylab="f(x)")
par(opar)
```

Index