Package: BayesianFactorZoo (via r-universe)

November 4, 2024

Title Bayesian Solutions for the Factor Zoo: We Just Ran Two Quadrillion Models

Version 0.0.0.3

Description Contains the functions to use the econometric methods in the paper Bryzgalova, Huang, and Julliard (2023) <doi:10.1111/jofi.13197>. In this package, we provide a novel Bayesian framework for analyzing linear asset pricing models: simple, robust, and applicable to high-dimensional problems. For a stand-alone model, we provide functions including BayesianFM() and BayesianSDF() to deliver reliable price of risk estimates for both tradable and nontradable factors. For competing factors and possibly nonnested models, we provide functions including continuous_ss_sdf(), continuous_ss_sdf_v2(), and dirac_ss_sdf_pvalue() to analyze high-dimensional models. If you use this package, please cite the paper. We are thankful to Yunan Ding and Jingtong Zhang for their research assistance. Any errors or omissions are the responsibility of the authors.

Imports MCMCpack, reshape2, MASS, timeSeries, coda, mvtnorm, matrixcalc, ggplot2, nse, Rdpack, stats, utils, Matrix

RdMacros Rdpack

Encoding UTF-8

RoxygenNote 7.3.1

License GPL-3

LazyData true

NeedsCompilation no

Author Svetlana Bryzgalova [aut], Jiantao Huang [cre], Christian Julliard [aut]

Maintainer Jiantao Huang <huangjt@hku.hk>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2024-10-04 09:30:08 UTC

Contents

BayesianFM 2
BayesianSDF 5
BFactor_zoo_example 9
continuous_ss_sdf 10
continuous_ss_sdf_v2 13
dirac_ss_sdf_pvalue 15
psi_to_priorSR 17
SDF_gmm 19
Two_Pass_Regression 21

Index **22**

`BayesianFM` *Bayesian Fama-MacBeth*

Description

This function provides the Bayesian Fama-MacBeth regression.

Usage

```
BayesianFM(f, R, sim_length)
```

Arguments

`f`	A matrix of factors with dimension $t \times k$, where k is the number of factors and t is the number of periods;
`R`	A matrix of test assets with dimension $t \times N$, where t is the number of periods and N is the number of test assets;
`sim_length`	The length of MCMCs;

Details

`BayesianFM` is similar to another twin function in this package, `BayesianSDF`, except that we estimate factors' risk premia rather than risk prices in this function. Unlike `BayesianSDF`, we use factor loadings, β_f, instead of covariance exposures, C_f, in the Fama-MacBeth regression. In particular, after we obtain the posterior draws of μ_Y and Σ_Y (details can be found in the section introducing `BayesianSDF` function), we calculate β_f as follows: $\beta_f = C_f \Sigma_f^{-1}$, and $\beta = (1_N, \beta_f)$.

Bayesian Fama-MacBeth (BFM)

The posterior distribution of λ conditional on μ_Y, Σ_Y, and the data, is a Dirac distribution at $(\beta^\top \beta)^{-1} \beta^\top \mu_R$.

Bayesian Fama-MacBeth GLS (BFM-GLS)

The posterior distribution of λ conditional on μ_Y, Σ_Y, and the data, is a Dirac distribution at $(\beta^\top \Sigma_R^{-1} \beta)^{-1} \beta^\top \Sigma_R^{-1} \mu_R$.

Value

The return of BayesianFM is a list of the following elements:

- lambda_ols_path: A sim_length$\times(k+1)$ matrix of OLS risk premia estimates (Each row represents a draw. Note that the first column is λ_c corresponding to the constant term. The next k columns are the risk premia estimates of the k factors);
- lambda_gls_path: A sim_length$\times(k+1)$ matrix of the risk premia estimates λ (GLS);
- R2_ols_path: A sim_length$\times 1$ matrix of the R^2_{OLS};
- R2_gls_path: A sim_length$\times 1$ matrix of the R^2_{GLS}.

Examples

```
## <------------------------------------------------------------------------------>
##   Example: Bayesian Fama-MacBeth
## <------------------------------------------------------------------------------>

library(reshape2)
library(ggplot2)

# Load Data
data("BFactor_zoo_example")
HML <- BFactor_zoo_example$HML
lambda_ols <- BFactor_zoo_example$lambda_ols
R2.ols.true <- BFactor_zoo_example$R2.ols.true
sim_f <- BFactor_zoo_example$sim_f
sim_R <- BFactor_zoo_example$sim_R
uf <- BFactor_zoo_example$uf

## <-------------------Case 1: strong factor---------------------------------------->

# the Frequentist Fama-MacBeth
# sim_f: simulated factor, sim_R: simulated return
# sim_f is the useful (i.e., strong) factor
results.fm <- Two_Pass_Regression(sim_f, sim_R)

# the Bayesian Fama-MacBeth with 10000 simulations
results.bfm <- BayesianFM(sim_f, sim_R, 2000)

# Note that the first element correspond to lambda of the constant term
# So we choose k=2 to get lambda of the strong factor
k <- 2
m1 <- results.fm$lambda[k]
sd1 <- sqrt(results.fm$cov_lambda[k,k])

bfm<-results.bfm$lambda_ols_path[1001:2000,k]
fm<-rnorm(20000,mean = m1, sd=sd1)
data<-data.frame(cbind(fm, bfm))
colnames(data)<-c("Frequentist FM", "Bayesian FM")
data.long<-melt(data)

p <- ggplot(aes(x=value, colour=variable, linetype=variable), data=data.long)
```

```
p+
 stat_density(aes(x=value, colour=variable),
              geom="line",position="identity", size = 2, adjust=1) +
 geom_vline(xintercept = lambda_ols[2], linetype="dotted", color = "#8c8c8c", size=1.5)+
 guides(colour = guide_legend(override.aes=list(size=2), title.position = "top",
 title.hjust = 0.5, nrow=1,byrow=TRUE))+
 theme_bw()+
 labs(color=element_blank()) +
 labs(linetype=element_blank()) +
 theme(legend.key.width=unit(4,"line")) +
 theme(legend.position="bottom")+
 theme(text = element_text(size = 26))+
 xlab(bquote("Risk premium ("~lambda[strong]~")")) +
 ylab("Density" )


## <-------------------Case 2: useless factor---------------------------------------->

# uf is the useless factor
# the Frequentist Fama-MacBeth
results.fm <- Two_Pass_Regression(uf, sim_R)

# the Bayesian Fama-MacBeth with 10000 simulations
results.bfm <- BayesianFM(uf, sim_R, 2000)

# Note that the first element correspond to lambda of the constant term
# So we choose k=2 to get lambda of the useless factor
k <- 2
m1 <- results.fm$lambda[k]
sd1 <- sqrt(results.fm$cov_lambda[k,k])


bfm<-results.bfm$lambda_ols_path[1001:2000,k]
fm<-rnorm(20000,mean = m1, sd=sd1)
data<-data.frame(cbind(fm, bfm))
colnames(data)<-c("Frequentist FM", "Bayesian FM")
data.long<-melt(data)

p <- ggplot(aes(x=value, colour=variable, linetype=variable), data=data.long)
p+
 stat_density(aes(x=value, colour=variable),
              geom="line",position="identity", size = 2, adjust=1) +
 geom_vline(xintercept = lambda_ols[2], linetype="dotted", color = "#8c8c8c", size=1.5)+
 guides(colour = guide_legend(override.aes=list(size=2),
 title.position = "top", title.hjust = 0.5, nrow=1,byrow=TRUE))+
 theme_bw()+
 labs(color=element_blank()) +
 labs(linetype=element_blank()) +
 theme(legend.key.width=unit(4,"line")) +
 theme(legend.position="bottom")+
 theme(text = element_text(size = 26))+
 xlab(bquote("Risk premium ("~lambda[strong]~")")) +
 ylab("Density" )
```

BayesianSDF *Bayesian estimation of Linear SDF (B-SDF)*

Description

This function provides the Bayesian estimates of factors' risk prices. The estimates with the flat prior are given by Definitions 1 and 2 in Bryzgalova et al. (2023). The estimates with the normal prior are used in Table I (see the footnote of Table I).

Usage

```
BayesianSDF(
  f,
  R,
  sim_length = 10000,
  intercept = TRUE,
  type = "OLS",
  prior = "Flat",
  psi0 = 5,
  d = 0.5
)
```

Arguments

`f`	A $t \times k$ matrix of factors, where k is the number of factors and t is the number of periods
`R`	A $t \times N$ matrix of test assets, where t is the number of periods and N is the number of test assets
`sim_length`	The length of MCMCs
`intercept`	If `intercept = TRUE` (`intercept = FALSE`), the model includes (does not include) the intercept. The default is `intercept = TRUE`
`type`	If `type = 'OLS'` (`type = 'GLS'`), the function returns Bayesian OLS (GLS) estimates of risk prices λ. The default is 'OLS'
`prior`	If `type = 'Flat'` (`type = 'Normal'`), the function executes the Bayesian estimation with the flat prior (normal prior). The default is 'Flat'
`psi0`	The hyper-parameter of the prior distribution of risk prices λ used in the normal prior (see **Details**). This parameter is needed only when the user chooses the normal prior. The default value is 5
`d`	The hyper-parameter of the prior distribution of risk prices λ used in the normal prior (see **Details**). The default value is 0.5

Details

Intercept

Consider the cross-sectional step. If one includes the intercept, the model is

$$\mu_R = \lambda_c 1_N + C_f \lambda_f = C\lambda,$$

where $C = (1_N, C_f)$ and $\lambda^\top = (\lambda_c^\top, \lambda_f^\top)^\top$. If one doesn't include the intercept, the model is

$$\mu_R = C_f \lambda_f = C\lambda,$$

where $C = C_f$ and $\lambda = \lambda_f$.

Bayesian Estimation

Let $Y_t = f_t \cup R_t$. Conditional on the data $Y = \{Y_t\}_{t=1}^T$, we can draw μ_Y and Σ_Y from the Normal-inverse-Wishart system

$$\mu_Y | \Sigma_Y, Y \sim N(\hat{\mu}_Y, \Sigma_Y / T),$$

$$\Sigma_Y | Y \sim W^{-1}(T-1, \Sigma_{t=1}^T (Y_t - \hat{\mu}_Y)(Y_t - \hat{\mu}_Y)^\top),$$

where W^{-1} is the inverse-Wishart distribution. We do not standardize Y_t in the time-series regression. In the empirical implementation, after obtaining posterior draws for μ_Y and Σ_Y, we calculate μ_R and C_f as the standardized expected returns of test assets and correlation between test assets and factors. It follows that C is a matrix containing a vector of ones and C_f.

The prior distribution of risk prices is either the flat prior or the normal prior.

With `prior = 'Flat'` and `type = 'OLS'`, for each draw, the risk price estimate is

$$\hat{\lambda} = (C^\top C)^{-1} C^T \mu_R.$$

With `prior = 'Flat'` and `type = 'GLS'`, for each draw, the risk price estimate is

$$\hat{\lambda} = (C^\top \Sigma_R^{-1} C)^{-1} C^\top \Sigma_R^{-1} \mu_R$$

If one chooses `prior = 'Normal'`, the prior of factor j's risk price is

$$\lambda_j | \sigma^2 \sim N(0, \sigma^2 \psi \tilde{\rho}_j^\top \tilde{\rho}_j T^d),$$

where $\tilde{\rho}_j = \rho_j - (\frac{1}{N} \Sigma_{i=1}^N \rho_{j,i}) \times 1_N$ is the cross-sectionally demeaned vector of factor j's correlations with asset returns. Equivalently,

$$\lambda | \sigma^2 \sim N(0, \sigma^2 D^{-1}),$$

$$D = diag\{(\psi \tilde{\rho}_1^\top \tilde{\rho}_1 T^d)^{-1}, ..., (\psi \tilde{\rho}_k^\top \tilde{\rho}_k T^d)^{-1}\} \ \ without\ intercept;$$

$$D = diag\{c, (\psi \tilde{\rho}_1^\top \tilde{\rho}_1 T^d)^{-1}, ..., (\psi \tilde{\rho}_k^\top \tilde{\rho}_k T^d)^{-1}\} \ \ with\ intercept;$$

where c is a small positive number corresponding to the common cross-sectional intercept (λ_c). Default values for ψ (`psi0`) and d (`d`) are 5 and 0.5, respectively.

With `prior = 'Normal'` and `type = 'OLS'`, for each draw, the risk price estimate is

$$\hat{\lambda} = (C^\top C + D)^{-1} C^\top \mu_R.$$

With `prior = 'Normal'` and `type = 'GLS'`, for each draw, the risk price estimate is

$$\hat{\lambda} = (C^\top \Sigma_R^{-1} C + D)^{-1} C^\top \Sigma_R^{-1} \mu_R.$$

Value

The return of BayesianSDF is a list that contains the following elements:

- lambda_path: A sim_length$\times(k+1)$ matrix if the intercept is included. NOTE: the first column λ_c corresponds to the intercept. The next k columns (i.e., the 2th – $(k+1)$-th columns) are the risk prices of k factors. If the intercept is excluded, the dimension of lambda_path is sim_length$\times k$.
- R2_path: A sim_length$\times 1$ matrix, which contains the posterior draws of the OLS or GLS R^2.

References

Bryzgalova S, Huang J, Julliard C (2023). "Bayesian solutions for the factor zoo: We just ran two quadrillion models <https://doi.org/10.1111/jofi.13197>." *Journal of Finance*, **78**(1), 487–557.

Examples

```
## <------------------------------------------------------------------------------>
## Example: Bayesian estimates of risk prices and R2
## This example is from the paper (see Section III. Simulation)
## <------------------------------------------------------------------------------>

library(reshape2)
library(ggplot2)

# Load the example data
data("BFactor_zoo_example")
HML <- BFactor_zoo_example$HML
lambda_ols <- BFactor_zoo_example$lambda_ols
R2.ols.true <- BFactor_zoo_example$R2.ols.true
sim_f <- BFactor_zoo_example$sim_f
sim_R <- BFactor_zoo_example$sim_R
uf <- BFactor_zoo_example$uf
W_ols <- BFactor_zoo_example$W_ols

cat("Load the simulated example \n")

cat("Cross-section: Fama-French 25 size and value portfolios \n")
cat("True pricing factor in simulations: HML \n")
cat("Pseudo-true cross-sectional R-squared:", R2.ols.true, "\n")
cat("Pseudo-true (monthly) risk price:", lambda_ols[2], "\n")

cat("----------------------------- Bayesian SDF ----------------------------\n")
cat("------------------------ See definitions 1 and 2 ----------------------\n")

cat("--------------------- Bayesian SDF: Strong factor ---------------------\n")

sim_result <- SDF_gmm(sim_R, sim_f, W_ols)   # GMM estimation
# sim_result$lambda_gmm
# sqrt(sim_result$Avar_hat[2,2])
# sim_result$R2_adj
```

```
## Now estimate the model using Bayesian method
two_step <- BayesianSDF(sim_f, sim_R, sim_length =  2000, psi0 = 5, d = 0.5)
# apply(X = two_step$lambda_path, FUN = quantile, MARGIN = 2, probs = c(0.05, 0.95))
# quantile(two_step$R2_path, probs = c(0.05, 0.5, 0.95))

# Note that the first element correspond to lambda of the constant term
# So we choose k=2 to get lambda of the strong factor
k <- 2
m1 <- sim_result$lambda_gmm[k]
sd1 <- sqrt(sim_result$Avar_hat[k,k])

bfm<-two_step$lambda_path[1001:2000, k]
fm<-rnorm(5000,mean = m1, sd=sd1)
data<-data.frame(cbind(fm, bfm))
colnames(data)<-c("GMM-OLS", "BSDF-OLS")
data.long<-melt(data)

#
### Figure 1(c)
#
p <- ggplot(aes(x=value, colour=variable, linetype=variable), data=data.long)
p+
 stat_density(aes(x=value, colour=variable),
              geom="line",position="identity", size = 2, adjust=1) +
 geom_vline(xintercept = lambda_ols[2], linetype="dotted", color = "#8c8c8c", size=1.5)+
 guides(colour = guide_legend(override.aes=list(size=2), title.position = "top",
 title.hjust = 0.5, nrow=1,byrow=TRUE))+
 theme_bw()+
 labs(color=element_blank()) +
 labs(linetype=element_blank()) +
 theme(legend.key.width=unit(4,"line")) +
 theme(legend.position="bottom")+
 theme(text = element_text(size = 26))+
 xlab(bquote("Risk price ("~lambda[strong]~")")) +
 ylab("Density" )


cat("--------------------- Bayesian SDF: Useless factor --------------------\n")

sim_result <- SDF_gmm(sim_R, uf, W_ols)
# sim_result$lambda_gmm
# sqrt(sim_result$Avar_hat[2,2])
# sim_result$R2_adj

two_step <- BayesianSDF(uf, sim_R, sim_length =  2000, psi0 = 5, d = 0.5)
#apply(X = two_step$lambda_path, FUN = quantile, MARGIN = 2, probs = c(0.05, 0.95))


## Posterior (Asymptotic) Distribution of lambda
k <- 2
m1 <- sim_result$lambda[k]
sd1 <- sqrt(sim_result$Avar_hat[k,k])
```

```
bfm<-two_step$lambda_path[1001:2000, k]
fm<-rnorm(5000,mean = m1, sd=sd1)
data<-data.frame(cbind(fm, bfm))
colnames(data)<-c("GMM-OLS", "BSDF-OLS")
data.long<-melt(data)

#
### Figure 1(a)
#
p <- ggplot(aes(x=value, colour=variable, linetype=variable), data=data.long)
p+
 stat_density(aes(x=value, colour=variable),
              geom="line",position="identity", size = 2, adjust=2) +
 geom_vline(xintercept = 0, linetype="dotted", color = "#8c8c8c", size=1.5)+
 guides(colour = guide_legend(override.aes=list(size=2),
 title.position = "top", title.hjust = 0.5, nrow=1,byrow=TRUE))+
 theme_bw()+
 labs(color=element_blank()) +
 labs(linetype=element_blank()) +
 theme(legend.key.width=unit(4,"line")) +
 theme(legend.position="bottom")+
 theme(text = element_text(size = 26))+
 xlab(bquote("Risk price ("~lambda[spurious]~")")) +
 ylab("Density" )
```

BFactor_zoo_example *Simulated Example Dataset* ***'BFactor_zoo_example'***

Description

A simulated dataset used in Figure 1 of Bryzgalova et al. (2023).

Usage

```
data("BFactor_zoo_example")
```

Format

A list consisting of the following variables:

HML High-minus-low value factor, from Ken French Website

lambda_ols Hypothetical true risk prices of factors in simulations

R2.ols.true Hypothetical true OLS R-squared in simulations

sim_f Simulated strong factor

sim_R Simulated test asset returns

uf Simulated weak/unspanned factor

W_ols Weighting matrix used in GMM OLS estimations

Source

Section III in Bryzgalova et al. (2023).

References

Bryzgalova S, Huang J, Julliard C (2023). "Bayesian solutions for the factor zoo: We just ran two quadrillion models <https://doi.org/10.1111/jofi.13197>." *Journal of Finance*, **78**(1), 487–557.

Examples

```
data(BFactor_zoo_example)
HML <- BFactor_zoo_example$HML
lambda_ols <- BFactor_zoo_example$lambda_ols
R2.ols.true <- BFactor_zoo_example$R2.ols.true
sim_f <- BFactor_zoo_example$sim_f
sim_R <- BFactor_zoo_example$sim_R
uf <- BFactor_zoo_example$uf
W_ols <- BFactor_zoo_example$W_ols

cat("Load the simulated example \n")
cat("Cross-section: Fama-French 25 size and value portfolios \n")
cat("True pricing factor in simulations: HML \n")
cat("Misspecified model with pseudo-true R-squared:", R2.ols.true, "\n")
cat("Pseudo-true (monthly) risk price:", lambda_ols[2], "\n")
```

---

continuous_ss_sdf *SDF model selection with continuous spike-and-slab prior*

Description

This function provides the SDF model selection procedure using the continuous spike-and-slab prior. See Propositions 3 and 4 in Bryzgalova et al. (2023).

Usage

```
continuous_ss_sdf(
  f,
  R,
  sim_length,
  psi0 = 1,
  r = 0.001,
  aw = 1,
  bw = 1,
  type = "OLS",
  intercept = TRUE
)
```

Arguments

`f`	A matrix of factors with dimension $t \times k$, where k is the number of factors and t is the number of periods;
`R`	A matrix of test assets with dimension $t \times N$, where t is the number of periods and N is the number of test assets;
`sim_length`	The length of monte-carlo simulations;
`psi0`	The hyper-parameter in the prior distribution of risk prices (see **Details**);
`r`	The hyper-parameter related to the prior of risk prices (see **Details**);
`aw`	The hyper-parameter related to the prior of γ (see **Details**);
`bw`	The hyper-parameter related to the prior of γ (see **Details**);
`type`	If `type = 'OLS'` (`type = 'GLS'`), the function returns Bayesian OLS (GLS) estimates of risk prices. The default is 'OLS'.
`intercept`	If `intercept = TRUE` (`intercept = FALSE`), we include (exclude) the common intercept in the cross-sectional regression. The default is `intercept = TRUE`.

Details

To model the variable selection procedure, we introduce a vector of binary latent variables $\gamma^\top = (\gamma_0, \gamma_1, ..., \gamma_K)$, where $\gamma_j \in \{0, 1\}$. When $\gamma_j = 1$, factor j (with associated loadings C_j) should be included in the model and vice verse.

The continuous spike-and-slab prior of risk prices λ is

$$\lambda_j | \gamma_j, \sigma^2 \sim N(0, r(\gamma_j)\psi_j\sigma^2).$$

When the factor j is included, we have $r(\gamma_j = 1) = 1$. When the factor is excluded from the model, $r(\gamma_j = 0) = r \ll 1$. Hence, the Dirac "spike" is replaced by a Gaussian spike, which is extremely concentrated at zero (the default value for r is 0.001). If `intercept = TRUE`, we choose $\psi_j = \psi\tilde{\rho}_j^\top \tilde{\rho}_j$, where $\tilde{\rho}_j = \rho_j - (\frac{1}{N}\Sigma_{i=1}^N \rho_{j,i}) \times 1_N$ is the cross-sectionally demeaned vector of factor j's correlations with asset returns. Instead, if `intercept = FALSE`, we choose $\psi_j = \psi\rho_j^\top \rho_j$. In the codes, ψ is equal to the value of `psi0`.

The prior $\pi(\omega)$ encoded the belief about the sparsity of the true model using the prior distribution $\pi(\gamma_j = 1|\omega_j) = \omega_j$. Following the literature on the variable selection, we set

$$\pi(\gamma_j = 1|\omega_j) = \omega_j, \;\; \omega_j \sim Beta(a_\omega, b_\omega).$$

Different hyperparameters a_ω and b_ω determine whether one a priori favors more parsimonious models or not. We choose $a_\omega = 1$ (`aw`) and $b_\omega = 1$ (`bw`) as the default values.

For each posterior draw of factors' risk prices $\lambda_f^{(j)}$, we can define the SDF as $m_t^{(j)} = 1 - (f_t - \mu_f)^\top \lambda_f^{(j)}$.The Bayesian model averaging of the SDF (BMA-SDF) over J draws is

$$m_t^{bma} = \frac{1}{J}\sum_{j=1}^{J} m_t^{(j)}.$$

Value

The return of `continuous_ss_sdf` is a list of the following elements:

- `gamma_path`: A `sim_length`$\times k$ matrix of the posterior draws of γ. Each row represents a draw. If $\gamma_j = 1$ in one draw, factor j is included in the model in this draw and vice verse.
- `lambda_path`: A `sim_length`$\times(k+1)$ matrix of the risk prices λ if `intercept = TRUE`. Each row represents a draw. Note that the first column is λ_c corresponding to the constant term. The next k columns (i.e., the 2-th – $(k+1)$-th columns) are the risk prices of the k factors. If `intercept = FALSE`, `lambda_path` is a `sim_length`$\times k$ matrix of the risk prices, without the estimates of λ_c.
- `sdf_path`: A `sim_length`$\times t$ matrix of posterior draws of SDFs. Each row represents a draw.
- `bma_sdf`: BMA-SDF.

References

Bryzgalova S, Huang J, Julliard C (2023). "Bayesian solutions for the factor zoo: We just ran two quadrillion models <https://doi.org/10.1111/jofi.13197>." *Journal of Finance*, **78**(1), 487–557.

Examples

```
## Load the example data
data("BFactor_zoo_example")
HML <- BFactor_zoo_example$HML
lambda_ols <- BFactor_zoo_example$lambda_ols
R2.ols.true <- BFactor_zoo_example$R2.ols.true
sim_f <- BFactor_zoo_example$sim_f
sim_R <- BFactor_zoo_example$sim_R
uf <- BFactor_zoo_example$uf

## sim_f: simulated strong factor
## uf: simulated useless factor

psi_hat <- psi_to_priorSR(sim_R, cbind(sim_f,uf), priorSR=0.1)
shrinkage <- continuous_ss_sdf(cbind(sim_f,uf), sim_R, 5000, psi0=psi_hat, r=0.001, aw=1, bw=1)
cat("Null hypothesis: lambda =", 0, "for each factor", "\n")
cat("Posterior probabilities of rejecting the above null hypotheses are:",
    colMeans(shrinkage$gamma_path), "\n")

## We also have the posterior draws of SDF: m(t) = 1 - lambda_g %*% (f(t) - mu_f)
sdf_path <- shrinkage$sdf_path

## We also provide the Bayesian model averaging of the SDF (BMA-SDF)
bma_sdf <- shrinkage$bma_sdf
```

`continuous_ss_sdf_v2` *SDF model selection with continuous spike-and-slab prior (tradable factors are treated as test assets)*

Description

This function provides the SDF model selection procedure using the continuous spike-and-slab prior. See Propositions 3 and 4 in Bryzgalova et al. (2023). Unlike `continuous_ss_sdf`, tradable factors are treated as test assets in this function.

Usage

```
continuous_ss_sdf_v2(
  f1,
  f2,
  R,
  sim_length,
  psi0 = 1,
  r = 0.001,
  aw = 1,
  bw = 1,
  type = "OLS",
  intercept = TRUE
)
```

Arguments

`f1`	A matrix of nontradable factors with dimension $t \times k_1$, where k_1 is the number of nontradable factors and t is the number of periods.
`f2`	A matrix of tradable factors with dimension $t \times k_2$, where k_2 is the number of tradable factors and t is the number of periods.
`R`	A matrix of test assets with dimension $t \times N$, where t is the number of periods and N is the number of test assets (`R` **should NOT contain tradable factors** `f2`);
`sim_length`	The length of monte-carlo simulations;
`psi0`	The hyper-parameter in the prior distribution of risk prices (see **Details**);
`r`	The hyper-parameter related to the prior of risk prices (see **Details**);
`aw`	The hyper-parameter related to the prior of γ (see **Details**);
`bw`	The hyper-parameter related to the prior of γ (see **Details**);
`type`	If `type = 'OLS'` (`type = 'GLS'`), the function returns Bayesian OLS (GLS) estimates of risk prices. The default is 'OLS'.
`intercept`	If `intercept = TRUE` (`intercept = FALSE`), we include (exclude) the common intercept in the cross-sectional regression. The default is `intercept = TRUE`.

Details

See the description in the twin function `continuous_ss_sdf`.

Value

The return of `continuous_ss_sdf_v2` is a list of the following elements:

- `gamma_path`: A `sim_length`$\times k$ matrix of the posterior draws of γ ($k = k_1 + k_2$). Each row represents a draw. If $\gamma_j = 1$ in one draw, factor j is included in the model in this draw and vice verse.
- `lambda_path`: A `sim_length`$\times(k+1)$ matrix of the risk prices λ if `intercept = TRUE`. Each row represents a draw. Note that the first column is λ_c corresponding to the constant term. The next k columns (i.e., the 2-th – $(k+1)$-th columns) are the risk prices of the k factors. If `intercept = FALSE`, `lambda_path` is a `sim_length`$\times k$ matrix of the risk prices, without the estimates of λ_c.
- `sdf_path`: A `sim_length`$\times t$ matrix of posterior draws of SDFs. Each row represents a draw.
- `bma_sdf`: BMA-SDF.

References

Bryzgalova S, Huang J, Julliard C (2023). "Bayesian solutions for the factor zoo: We just ran two quadrillion models <https://doi.org/10.1111/jofi.13197>." *Journal of Finance*, **78**(1), 487–557.

Examples

```
library(timeSeries)

## Load the example data
data("BFactor_zoo_example")
HML <- BFactor_zoo_example$HML
lambda_ols <- BFactor_zoo_example$lambda_ols
R2.ols.true <- BFactor_zoo_example$R2.ols.true
sim_f <- BFactor_zoo_example$sim_f
sim_R <- BFactor_zoo_example$sim_R
uf <- BFactor_zoo_example$uf

## sim_f: simulated strong factor
## uf: simulated useless factor

psi_hat <- psi_to_priorSR(sim_R, cbind(sim_f,uf,sim_R[,1]), priorSR=0.1)

## We include the first test asset, sim_R[,1], into factors, so f2 = sim_R[,1,drop=FALSE].
## Also remember excluding sim_R[,1,drop=FALSE] from test assets, so R = sim_R[,-1].
shrinkage <- continuous_ss_sdf_v2(cbind(sim_f,uf), sim_R[,1,drop=FALSE], sim_R[,-1], 1000,
                                  psi0=psi_hat, r=0.001, aw=1, bw=1)
cat("Null hypothesis: lambda =", 0, "for each of these three factors", "\n")
cat("Posterior probabilities of rejecting the above null hypotheses are:",
    colMeans(shrinkage$gamma_path), "\n")

## We also have the posterior draws of SDF: m(t) = 1 - lambda_g %*% (f(t) - mu_f)
```

```
sdf_path <- shrinkage$sdf_path

## We also provide the Bayesian model averaging of the SDF (BMA-SDF)
bma_sdf <- shrinkage$bma_sdf

## We can further estimate the posterior distributions of model-implied Sharpe ratios:
cat("The 5th, 50th, and 95th quantiles of model-implied Sharpe ratios:",
    quantile(colSds(t(sdf_path)), probs=c(0.05, 0.5, 0.95)), "\n")

## Finally, we can estimate the posterior distribution of model dimensions:
cat("The posterior distribution of model dimensions (= 0, 1, 2, 3):",
    prop.table(table(rowSums(shrinkage$gamma_path))), "\n")

## We now use the 17th test asset, sim_R[,17,drop=FALSE], as the tradable factor,
## so f2 = sim_R[,17,drop=FALSE].
## Also remember excluding sim_R[,17,drop=FALSE] from test assets, so R = sim_R[,-17].
psi_hat <- psi_to_priorSR(sim_R, cbind(sim_f,uf,sim_R[,17]), priorSR=0.1)
shrinkage <- continuous_ss_sdf_v2(cbind(sim_f,uf), sim_R[,17,drop=FALSE], sim_R[,-17],
                                  1000, psi0=psi_hat, r=0.001, aw=1, bw=1)
cat("Null hypothesis: lambda =", 0, "for each of these three factors", "\n")
cat("Posterior probabilities of rejecting the above null hypotheses are:",
    colMeans(shrinkage$gamma_path), "\n")
```

`dirac_ss_sdf_pvalue` *Hypothesis testing for risk prices (Bayesian p-values) with Dirac spike-and-slab prior*

Description

This function tests the null hypothesis, $H_0 : \lambda = \lambda_0$, when $\gamma = 0$. When $\lambda_0 = 0$, we compare factor models using the algorithm in Proposition 1 of Bryzgalova et al. (2023). When $\lambda_0 \neq 0$, this function corresponds to Corollary 2 in Section II.A.2 of Bryzgalova et al. (2023). The function can also be used to compute the posterior probabilities of all possible models with up to a given maximum number of factors (see examples).

Usage

```
dirac_ss_sdf_pvalue(f, R, sim_length, lambda0, psi0 = 1, max_k = NULL)
```

Arguments

`f`	A matrix of factors with dimension $t \times k$, where k is the number of factors and t is the number of periods;
`R`	A matrix of test assets with dimension $t \times N$, where t is the number of periods and N is the number of test assets;
`sim_length`	The length of Monte-Carlo simulations;

`lambda0`	A $k \times 1$ vector of risk prices under the null hypothesis ($\gamma = 0$);
`psi0`	The hyper-parameter in the prior distribution of risk price λ (see **Details**);
`max_k`	The maximal number of factors in models (`max_k` is a positive integer or `NULL` if the user does not impose any restriction on the model dimension).

Details

Let D denote a diagonal matrix with elements $c, \psi_1^{-1}, ..., \psi_K^{-1}$, and D_γ the submatrix of D corresponding to model γ, where c is a small positive number corresponding to the common cross-sectional intercept (λ_c). The prior for the prices of risk (λ_γ) of model γ is then

$$\lambda_\gamma | \sigma^2, \gamma \sim N(0, \sigma^2, D_\gamma^{-1}).$$

We choose $\psi_j = \psi \tilde{\rho}_j^\top \tilde{\rho}_j$, where $\tilde{\rho}_j = \rho_j - (\frac{1}{N}\Sigma_{i=1}^N \rho_{j,i}) \times 1_N$ is the cross-sectionally demeaned vector of factor j's correlations with asset returns. In the codes, ψ is equal to the value of `psi0`.

Value

The return of `dirac_ss_sdf_pvalue` is a list of the following elements:

- `gamma_path`: A `sim_length`$\times k$ matrix of the posterior draws of γ. Each row represents a draw. If $\gamma_j = 1$ in one draw, factor j is included in the model in this draw and vice verse.
- `lambda_path`: A `sim_length`$\times (k+1)$ matrix of the risk prices λ. Each row represents a draw. Note that the first column is λ_c corresponding to the constant term. The next k columns (i.e., the 2-th – $(k+1)$-th columns) are the risk prices of the k factors;
- `model_probs`: A $2^k \times (k+1)$ matrix of posterior model probabilities, where the first k columns are the model indices and the final column is a vector of model probabilities.

References

Bryzgalova S, Huang J, Julliard C (2023). "Bayesian solutions for the factor zoo: We just ran two quadrillion models <https://doi.org/10.1111/jofi.13197>." *Journal of Finance*, **78**(1), 487–557.

Examples

```
## <------------------------------------------------------------------------------>
## Example: Bayesian p-value (with the dirac spike-and-slab prior)
## <------------------------------------------------------------------------------>

# Load the example data
data("BFactor_zoo_example")
HML <- BFactor_zoo_example$HML
lambda_ols <- BFactor_zoo_example$lambda_ols
R2.ols.true <- BFactor_zoo_example$R2.ols.true
sim_f <- BFactor_zoo_example$sim_f
sim_R <- BFactor_zoo_example$sim_R
uf <- BFactor_zoo_example$uf

### Now we estimate the Bayesian p-values defined in Corollary 2.
```

```
#
### Prior Sharpe ratio of factor model for different values of psi: see equation (27):
#
cat("--------------- Choose psi based on prior Sharpe ratio ----------------\n")
cat("if psi = 1, prior Sharpe ratio is", psi_to_priorSR(sim_R, sim_f, psi0=1), "\n")
cat("if psi = 2, prior Sharpe ratio is", psi_to_priorSR(sim_R, sim_f, psi0=2), "\n")
cat("if psi = 5, prior Sharpe ratio is", psi_to_priorSR(sim_R, sim_f, psi0=5), "\n")

## Test whether factors' risk prices equal 'matrix(lambda_ols[2]*sd(HML),ncol=1)'
## Bayesian p-value is given by mean(shrinkage$gamma_path)
shrinkage <- dirac_ss_sdf_pvalue(sim_f, sim_R, 1000, matrix(lambda_ols[2]*sd(HML),ncol=1))
cat("Null hypothesis: lambda =", matrix(lambda_ols[2]*sd(HML)), "\n")
cat("Posterior probability of rejecting the above null hypothesis is:",
    mean(shrinkage$gamma_path), "\n")

## Test whether the risk price of factor 'sim_f' is equal to 0
shrinkage <- dirac_ss_sdf_pvalue(sim_f, sim_R, 1000, 0, psi0=1)
cat("Null hypothesis: lambda =", 0, "\n")
cat("Posterior probability of rejecting the above null hypothesis is:",
    mean(shrinkage$gamma_path), "\n")


## One can also put more than one factor into the test
two_f = cbind(sim_f,uf) # sim_f is the strong factor while uf is the useless factor
# Test1: lambda of sim_f = 0, Test2: lambda of uf = 0
lambda0_null_vec = t(cbind(0,0)) # 2x1 vector
shrinkage <- dirac_ss_sdf_pvalue(two_f, sim_R, 1000, lambda0_null_vec, psi0=1)
cat("Null hypothesis: lambda =", 0, "for each factor", "\n")
cat("Posterior probabilities of rejecting the above null hypothesis are:",
    colMeans(shrinkage$gamma_path), "\n")

## We can also print the posterior model probabilities:
cat('Posterior model probabilities are:\n')
print(shrinkage$model_probs)


## One can compute the posterior probabilities of all possible models with up to
## a given maximum number of factors. For example, we consider two factors, but
## the number of factors is restricted to be less than two.
lambda0_null_vec = t(cbind(0,0)) # 2x1 vector
shrinkage <- dirac_ss_sdf_pvalue(two_f, sim_R, 1000, lambda0_null_vec, psi0=1, max_k=1)
cat('Posterior model probabilities are:\n')
print(shrinkage$model_probs)
## Comment: You may notice that the model with index (1, 1) has a posterior probability
##          of exactly zero since the maximal number of factors is one.
```

`psi_to_priorSR` *Mapping ψ (`psi0`) to the prior Sharpe ratio of factors (`priorSR`), and vice versa.*

Description

This function provides the one-to-one mapping between ψ and the prior Sharpe ratio of factors. See Section II.A.3 in Bryzgalova et al. (2023).

Usage

```
psi_to_priorSR(R, f, psi0 = NULL, priorSR = NULL, aw = 1, bw = 1)
```

Arguments

R	A matrix of test assets with dimension $t \times N$, where t is the number of periods and N is the number of test assets;
f	A matrix of factors with dimension $t \times k$, where k is the number of factors and t is the number of periods;
psi0	The hyper-parameter in the prior distribution of risk prices (see **Details** in the function `continuous_ss_sdf`);
priorSR	The prior Sharpe ratio of all factors (see **Details**);
aw	The hyper-parameter in the prior of γ (default value = 1, see **Details**);
bw	The hyper-parameter in the prior of γ (default value = 1, see **Details**);

Details

According to equation (27) in Bryzgalova et al. (2023), we learn that

$$\frac{E_\pi[SR_f^2 \mid \gamma, \sigma^2]}{E_\pi[SR_\alpha^2 \mid \sigma^2]} = \frac{\psi \sum_{k=1}^K r(\gamma_k) \tilde{\rho}_k^\top \tilde{\rho}_k}{N},$$

where SR_f^2 and SR_α^2 denote the Sharpe ratios of all factors (f_t) and of the pricing errors (α), and E_π denotes prior expectations.

The prior $\pi(\omega)$ encodes the belief about the sparsity of the true model using the prior distribution $\pi(\gamma_j = 1|\omega_j) = \omega_j, \;\; \omega_j \sim Beta(a_\omega, b_\omega)$. We further integrate out γ_j in $E_\pi[SR_f^2 \mid \gamma, \sigma^2]$ and show the following:

$$\frac{E_\pi[SR_f^2 \mid \sigma^2]}{E_\pi[SR_\alpha^2 \mid \sigma^2]} \approx \frac{a_\omega}{a_\omega + b_\omega} \psi \frac{\sum_{k=1}^K \tilde{\rho}_k^\top \tilde{\rho}_k}{N}, \; as \; r \to 0.$$

Since we can decompose the Sharpe ratios of all test assets, SR_R^2, into SR_f^2 and SR_α^2 (i.e., $SR_R^2 = SR_f^2 + SR_\alpha^2$), we can represent SR_f^2 as follows:

$$E_\pi[SR_f^2 \mid \sigma^2] \approx \frac{\frac{a_\omega}{a_\omega + b_\omega} \psi \frac{\sum_{k=1}^K \tilde{\rho}_k^\top \tilde{\rho}_k}{N}}{1 + \frac{a_\omega}{a_\omega + b_\omega} \psi \frac{\sum_{k=1}^K \tilde{\rho}_k^\top \tilde{\rho}_k}{N}} SR_R^2.$$

We define the prior Sharpe ratio implied by the factor models as $\sqrt{E_\pi[SR_f^2 \mid \sigma^2]}$. Given a_ω, b_ω, $\frac{\sum_{k=1}^K \tilde{\rho}_k^\top \tilde{\rho}_k}{N}$, and the observed Sharpe ratio of test assets, we have one-to-one mapping between ψ and $\sqrt{E_\pi[SR_f^2 \mid \sigma^2]}$.

If the user aims to convert ψ to the prior Sharpe ratio, she should input only `psi0`. In contrast, if she wants to convert the prior Sharpe ratio to ψ, `priorSR` should be entered.

Value

The return of `psi_to_priorSR` is:

- `psi0` or `priorSR`.

References

Bryzgalova S, Huang J, Julliard C (2023). "Bayesian solutions for the factor zoo: We just ran two quadrillion models <https://doi.org/10.1111/jofi.13197>." *Journal of Finance*, **78**(1), 487–557.

Examples

```
## Load the example data
data("BFactor_zoo_example")
HML <- BFactor_zoo_example$HML
lambda_ols <- BFactor_zoo_example$lambda_ols
R2.ols.true <- BFactor_zoo_example$R2.ols.true
sim_f <- BFactor_zoo_example$sim_f
sim_R <- BFactor_zoo_example$sim_R
uf <- BFactor_zoo_example$uf

## If the user aims to convert \eqn{\psi} to the prior Sharpe ratio:
print(psi_to_priorSR(sim_R, sim_f, priorSR=0.1))

## If the user  wants to convert the prior Sharpe ratio to \eqn{\psi}:
psi0_to_map <- psi_to_priorSR(sim_R, sim_f, priorSR=0.1)
print(psi_to_priorSR(sim_R, sim_f, psi0=psi0_to_map))

## If we enter both psi0 and priorSR (or forget to input them simultaneously),
## a warning will be printed:
print(psi_to_priorSR(sim_R, sim_f))
print(psi_to_priorSR(sim_R, sim_f, priorSR=0.1, psi0=2))
```

`SDF_gmm` *GMM Estimates of Factors' Risk Prices under the Linear SDF Framework*

Description

This function provides the GMM estimates of factors' risk prices under the linear SDF framework (including the common intercept).

Usage

```
SDF_gmm(R, f, W)
```

Arguments

R	A matrix of test assets with dimension $t \times N$, where t is the number of periods and N is the number of test assets;
f	A matrix of factors with dimension $t \times k$, where k is the number of factors and t is the number of periods;
W	Weighting matrix in GMM estimation (see **Details**).

Details

We follow the notations in Section I of Bryzgalova et al. (2023). Suppose that there are K factors, $f_t = (f_{1t}, ..., f_{Kt})^\top, t = 1, ..., T$. The returns of N test assets are denoted by $R_t = (R_{1t}, ..., R_{Nt})^\top$.

Consider linear SDFs (M), that is, models of the form $M_t = 1 - (f_t - E[f_t])^\top \lambda_f$.

The model is estimated via GMM with moment conditions

$$E[g_t(\lambda_c, \lambda_f, \mu_f)] = E\begin{pmatrix} R_t - \lambda_c 1_N - R_t(f_t - \mu_f)^\top \lambda_f \\ f_t - \mu_f \end{pmatrix} = \begin{pmatrix} 0_N \\ 0_K \end{pmatrix}$$

and the corresponding sample analog function $g_T(\lambda_c, \lambda_f, \mu_f) = \frac{1}{T}\Sigma_{t=1}^T g_t(\lambda_c, \lambda_f, \mu_f)$. Different weighting matrices deliver different point estimates. Two popular choices are

$$W_{ols} = \begin{pmatrix} I_N & 0_{N\times K} \\ 0_{K\times N} & \kappa I_K \end{pmatrix}, \ W_{gls} = \begin{pmatrix} \Sigma_R^{-1} & 0_{N\times K} \\ 0_{K\times N} & \kappa I_K \end{pmatrix},$$

where Σ_R is the covariance matrix of returns and $\kappa > 0$ is a large constant so that $\hat{\mu}_f = \frac{1}{T}\Sigma_{t=1}^T f_t$.

The asymptotic covariance matrix of risk premia estimates, `Avar_hat`, is based on the assumption that $g_t(\lambda_c, \lambda_f, \mu_f)$ is independent over time.

Value

The return of `SDF_gmm` is a list of the following elements:

- `lambda_gmm`: Risk price estimates;
- `mu_f`: Sample means of factors;
- `Avar_hat`: Asymptotic covariance matrix of GMM estimates (see **Details**);
- `R2_adj`: Adjusted cross-sectional R^2;
- `S_hat`: Spectral matrix.

References

Bryzgalova S, Huang J, Julliard C (2023). "Bayesian solutions for the factor zoo: We just ran two quadrillion models <https://doi.org/10.1111/jofi.13197>." *Journal of Finance*, **78**(1), 487–557.

`Two_Pass_Regression` *Fama MacBeth Two-Pass Regression*

Description

This function provides the frequentist Fama-MacBeth Two-Pass Regression.

Usage

```
Two_Pass_Regression(f, R)
```

Arguments

`f`	A matrix of factors with dimension $t \times k$, where k is the number of factors and t is the number of periods;
`R`	A matrix of test assets with dimension $t \times N$, where t is the number of periods and N is the number of test assets;

Details

See Chapter 12.2 in Cochrane (2009). `t_stat` and `t_stat_gls` are t-statistics of OLS and GLS risk premia estimates based on the asymptotic standard errors in equation (12.19) in Cochrane (2009).

Value

The return of `Two_Pass_Regression` is a list of the following elements:

- lambda: Risk premia estimates in the OLS two-pass regression;
- lambda_gls: Risk premia estimates in the GLS two-pass regression;
- t_stat: The t-statistics of risk premia estimates in the OLS two-pass regression;
- t_stat_gls: The t-statistics of risk premia estimates in the GLS two-pass regression;
- R2_adj: Adjusted $R2$ in the OLS two-pass regression;
- R2_adj_GLS: Adjusted $R2$ in the GLS two-pass regression.

References

Cochrane J (2009). *Asset pricing: Revised edition*. Princeton University Press.

Index

∗ **datasets**
BFactor_zoo_example, 9

BayesianFM, 2
BayesianSDF, 5
BFactor_zoo_example, 9

continuous_ss_sdf, 10
continuous_ss_sdf_v2, 13

dirac_ss_sdf_pvalue, 15

psi_to_priorSR, 17

SDF_gmm, 19

Two_Pass_Regression, 21