Package 'GARCHInfoLSTM'

September 21, 2026

Type Package

Title GARCH-Informed LSTM Model for Volatility Forecasting

Version 0.1.0

Description The proposed Generalized Autoregressive Conditional Heteroskedasticity (GARCH)-informed Long Short-Term Memory (LSTM) model follows the concept of physics-informed machine learning (PIML) by integrating established econometric knowledge of price volatility into a data-driven forecasting framework. In the model, conditional volatility estimated from the GARCH process is incorporated as an additional explanatory signal or volatility-based weighting component within the LSTM architecture. This enables the LSTM to learn nonlinear temporal dependencies while remaining informed by the underlying characteristics of agricultural price series, including volatility clustering, heteroscedasticity and market uncertainty. The optimized weighting parameter, lambda, controls the contribution of the GARCH-derived volatility information to the final prediction. Thus, the model combines the statistical interpretability of GARCH with the nonlinear learning capability of LSTM, producing a hybrid PIML framework that is more responsive to both normal price movements and periods of extreme market volatility. The methodology is motivated by hybrid forecasting framework proposed by Yeasin and Paul (2024) <doi:10.1007/s11227-023-05542-3>.

Encoding UTF-8

Imports torch (>= 0.11.0), rugarch (>= 1.5.0), ggplot2 (>= 3.4.0), cli (>= 3.6.0), coro, stats, utils

License MIT + file LICENSE

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Md Yeasin [aut],
Ranjit Kumar Paul [aut, cre],
Manojit Mandal [aut],
Pushkar Bora [aut]

Maintainer Ranjit Kumar Paul <ranjitstat@gmail.com>

Repository CRAN

Date/Publication 2026-09-21 20:50:08 UTC

Contents

`AccuracyGINN` *Out-of-Sample Accuracy for GINN Volatility Forecasts*

Description

Compares GINN conditional variance forecasts against realized variance computed from actual data that arrived after training. Returns RMSE, MAE, R-squared and SMAPE on variance units.

Usage

```
AccuracyGINN(fit, actual, lambda = NULL, verbose = TRUE)
```

Arguments

`fit`	A GINN object returned by GINN().
`actual`	Numeric vector of actual DATA observed after training. Their returns and realized variance are computed internally. Needs at least 2 values.
`lambda`	Lambda value to use. NULL = best lambda.
`verbose`	Logical. If TRUE (default), prints a formatted accuracy report to the console. Set to FALSE to suppress console output and only receive the returned data frame.

Value

Data frame with columns: Lambda, h, RMSE, MAE, R2, SMAPE (all in variance units). Also prints a formatted accuracy table to the console. The data frame has a "comparison" attribute with step-by-step Actual_Var vs Forecast_Var vs Error columns.

Examples

```
prices <- cumprod(1 + rnorm(60, 0.001, 0.02)) * 100
fit    <- GINN(prices, mode = "manual", lag = 5,
               ar_lag = 1, garch_p = 1, garch_q = 1,
               garch_mean = "constant", garch_dist = "norm",
               hidden_size = 4, num_layers = 1, lr = 0.01,
               epochs = 20, patience = 5,
               lambda_list = c(0.5), verbose = FALSE)
```

```
# Simulate actual future prices
new_prices <- tail(prices, 1) * cumprod(1 + rnorm(12, 0.001, 0.02))

acc <- AccuracyGINN(fit, actual = new_prices)
print(acc)

# Step-by-step comparison
attr(acc, "comparison")
```

GINN *GARCH-Informed Neural Network (GINN) – Volatility Forecasting*

Description

Trains an LSTM on conditional variance sequences using the GINN composite loss:

```
Loss = (1-lambda) x MSE(sigma2_t, sigma2_hat_LSTM)
     + lambda     x MSE(sigma2_GARCH, sigma2_hat_LSTM)
```

Trains on FULL data – no train/test split. Best lambda selected by Training RMSE. Use predictGINN() for out-of-sample volatility forecasting.

Pipeline:

1. Returns: $r_t = (y_t - y_{t-1})/y_{t-1}$
2. AR(p) on full returns gives mean return mu_t
3. GARCH(p,q) on full returns gives sigma2_GARCH
4. Ground-truth variance: sigma2_t = (r_t - mu_t)^2
5. LSTM trained on sigma2_t sequences with GINN loss

Usage

```
GINN(
  data,
  mode = c("auto", "manual"),
  ar_max_lag = 5L,
  ar_lag = NULL,
  garch_p_grid = 1:3,
  garch_q_grid = 1:3,
  garch_mean_grid = c("constant", "ar", "arma"),
  garch_dist_grid = c("norm", "std", "ged", "snorm", "sstd", "sged"),
  garch_p = NULL,
  garch_q = NULL,
  garch_mean = NULL,
  garch_dist = NULL,
  lag,
```

```
  hidden_size = 32,
  num_layers = 1,
  lr = 0.001,
  dropout = 0.0,
  epochs = 500,
  patience = 30,
  tune_epochs = 50,
  batch_size = 16,
  lambda_list = seq(0.1, 0.9, 0.1),
  hidden_grid = c(8, 16, 24, 32, 40),
  layers_grid = c(1, 2),
  lr_grid = c(0.001, 0.003, 0.005),
  dropout_grid = c(0.0, 0.1, 0.2, 0.3),
  verbose = TRUE,
  seed = 42L
)
```

Arguments

data	Numeric data vector. ALL observations used for training. GINN internally computes returns and conditional variance.
mode	"auto" for automatic AR + GARCH + LSTM tuning. "manual" for user-supplied parameters.
ar_max_lag	Max AR lag to search (auto mode). Default 5.
ar_lag	Fixed AR lag (manual mode). NULL = auto.
garch_p_grid	ARCH order grid (auto mode). Default 1:3.
garch_q_grid	GARCH order grid (auto mode). Default 1:3.
garch_mean_grid	Mean model candidates for grid search: "constant" (intercept only), "ar" (AR(p) mean), "arma" (ARMA mean). Default c("constant", "ar", "arma").
garch_dist_grid	Error distribution candidates: "norm" (normal), "std" (Student-t), "ged" (GED), "snorm" (skew-normal), "sstd" (skew-t), "sged" (skew-GED). Default includes all six.
garch_p	Fixed ARCH order (manual mode).
garch_q	Fixed GARCH order (manual mode).
garch_mean	Fixed mean model (manual mode): "constant", "ar", or "arma".
garch_dist	Fixed distribution (manual mode).
lag	Variance sequence look-back length. COMPULSORY – no default. User must specify. Example: lag = 5 uses the last 5 variance values to predict the next one.
hidden_size	LSTM hidden units (manual mode). Default 32.
num_layers	Stacked LSTM layers (manual mode). Default 1.
lr	Learning rate (manual mode). Default 0.001.
dropout	Dropout rate (manual mode). Default 0.0.

epochs	Max training epochs. Default 500.
patience	Early-stopping patience. Default 30.
tune_epochs	Epochs per grid-search combo (auto mode). Default 50.
batch_size	Mini-batch size. Default 16.
lambda_list	Lambda values to test. Default seq(0.1, 0.9, 0.1). lambda = 0.0: Standard LSTM (GT variance only, no GARCH guidance). lambda = 0.5: Balanced GINN (equal GT and GARCH weight). lambda = 1.0: GINN-1 (GARCH variance only).
hidden_grid	Hidden-unit candidates (auto mode).
layers_grid	Layer-count candidates (auto mode).
lr_grid	Learning-rate candidates (auto mode).
dropout_grid	Dropout candidates (auto mode).
verbose	Print progress messages. Default TRUE.
seed	Random seed. Default 42.

Value

S3 object of class "GINN" with elements:

results Named list per lambda. Each entry contains: tr_rmse, tr_mae, tr_r2, tr_preds, tr_actuals (all in VARIANCE units), val_rmse, loss_hist, time, lambda, model.

best_model Complete best lambda summary: lambda, hyperparams, ar_lag, garch_config, metrics table, predictions (Actual_Var vs Predicted_Var data frame), loss_history, model object, train_time.

garch GARCH config, sigma2_all (full in-sample variance), coefs.

ar AR lag and fitted mean returns mu_all.

returns Full return series r_t.

variance Ground-truth variance sigma2_t (all observations).

best_hp Best LSTM hyperparameters found.

meta Settings: lag, lambda_list, mode, best_lam_nm, selection = "Train RMSE".

data_info prices, N, N_ret, var_scaler, last_var_seq_sc, lag, mu_all.

Examples

```
prices <- cumprod(1 + rnorm(60, 0.001, 0.02)) * 100

# -- Fast manual mode (fixed GARCH order + small LSTM) -------
fit <- GINN(prices, mode = "manual", lag = 5,
            ar_lag = 1, garch_p = 1, garch_q = 1,
            garch_mean = "constant", garch_dist = "norm",
            hidden_size = 4, num_layers = 1, lr = 0.01,
            epochs = 20, patience = 5,
            lambda_list = c(0.5), verbose = FALSE)
print(fit)
summary(fit)
```

```
# -- Forecast 12 steps of conditional variance --------------
fc <- predictGINN(fit, h = 12)
print(fc)

# Volatility = sqrt(variance)
sqrt(fc$forecast_var)

# -- Diagnostic plots ---------------------------------------
plotGINN(fit, h = 12)      # actual + predicted + forecast
plotlambda(fit)             # Training RMSE by lambda

# -- Best model details -------------------------------------
fit$best_model
fit$best_model$lambda
fit$best_model$hyperparams
fit$best_model$metrics
fit$best_model$predictions   # Actual_Var vs Predicted_Var

# -- Manual mode with a couple of lambdas -------------------
fit2 <- GINN(
  prices,
  mode       = "manual",
  lag        = 5,
  ar_lag     = 1,
  garch_p    = 1,
  garch_q    = 1,
  garch_mean = "ar",
  garch_dist = "norm",
  hidden_size = 4,
  num_layers  = 1,
  lr          = 0.01,
  epochs      = 20,
  patience    = 5,
  dropout     = 0.0,
  lambda_list = c(0.3, 0.7),
  verbose     = FALSE
)
summary(fit2)

# -- Accuracy vs actual future prices ----------------------
new_prices <- tail(prices, 1) * cumprod(1 + rnorm(12, 0.001, 0.02))
AccuracyGINN(fit, actual = new_prices)

# Plot with actual future variance overlaid in green
plotGINN(fit, h = 12, actual = new_prices)
```

---

plotGINN *Plot GINN Volatility: Actual vs Predicted + Forecast*

Description

Plots conditional variance (volatility) in one chart showing:

1. Actual realized variance sigma2_t on full training data (dark line).
2. GINN in-sample predicted variance (blue line).
3. Out-of-sample variance forecast (red solid line).

If actual future prices are provided, their realized variance is computed and overlaid as a green line for direct comparison with the red forecast.

Usage

```
plotGINN(fit, h = 12, lambda = NULL, actual = NULL,
          newdata = NULL, n_hist = NULL)
```

Arguments

fit	A GINN object returned by GINN().
h	Forecast horizon (steps ahead in variance space). Default 12.
lambda	Lambda to use. NULL = best lambda.
actual	Optional numeric vector of actual future PRICES. Their variance is computed and overlaid in green.
newdata	Optional new prices to update starting sequence.
n_hist	Number of historical variance points to show. NULL = all.

Value

A ggplot2 object (invisibly).

Examples

```
prices <- cumprod(1 + rnorm(60, 0.001, 0.02)) * 100
fit    <- GINN(prices, mode = "manual", lag = 5,
               ar_lag = 1, garch_p = 1, garch_q = 1,
               garch_mean = "constant", garch_dist = "norm",
               hidden_size = 4, num_layers = 1, lr = 0.01,
               epochs = 20, patience = 5,
               lambda_list = c(0.5), verbose = FALSE)

# Plot variance forecast only
plotGINN(fit, h = 12)

# Show last 50 historical variance points only
plotGINN(fit, h = 12, n_hist = 50)

# With actual future prices overlaid (green = actual variance)
new_prices <- tail(prices, 1) * cumprod(1 + rnorm(12, 0.001, 0.02))
plotGINN(fit, h = 12, actual = new_prices)
```

plotlambda *Plot Training RMSE Across Lambda Values (GINN)*

Description

Plot Training RMSE Across Lambda Values (GINN)

Usage

```
plotlambda(fit)
```

Arguments

fit	A GINN object.

Value

ggplot2 object (invisibly).

predictGINN *Forecast Conditional Variance h Steps Ahead from GINN*

Description

Forecasts future conditional variance (volatility) h steps ahead using recursive one-step prediction on variance sequences. Returns single-point forecasts only.

Usage

```
predictGINN(fit, h = 1, lambda = NULL, newdata = NULL)
```

Arguments

fit	A GINN object returned by GINN().
h	Number of steps to forecast ahead. Default 1.
lambda	Lambda value to use. NULL = best (lowest Train RMSE). Use 0.0 for Standard LSTM (GT variance only, no GARCH guidance).
newdata	Optional numeric vector of new DATA that arrived after training. Their variance is computed and used to update the starting sequence before forecasting.

Value

Data frame with columns:

h Forecast horizon step (1, 2, ..., h).

time_idx Time index in return space (N_ret + 1, ..., N_ret + h).

forecast_var Point forecast of conditional variance sigma2_t.

Examples

```
prices <- cumprod(1 + rnorm(60, 0.001, 0.02)) * 100
fit    <- GINN(prices, mode = "manual", lag = 5,
                ar_lag = 1, garch_p = 1, garch_q = 1,
                garch_mean = "constant", garch_dist = "norm",
                hidden_size = 4, num_layers = 1, lr = 0.01,
                epochs = 20, patience = 5,
                lambda_list = c(0.3, 0.7), verbose = FALSE)

# Forecast 12 steps of conditional variance
fc <- predictGINN(fit, h = 12)
print(fc)

# Volatility = sqrt(variance)
sqrt(fc$forecast_var)

# Specific lambda
fc3 <- predictGINN(fit, h = 6, lambda = 0.3)

# With new prices to update starting sequence
new_prices <- tail(prices, 1) * cumprod(1 + rnorm(5, 0.001, 0.02))
fc_new     <- predictGINN(fit, h = 12, newdata = new_prices)
```

Index