# A New (Old) Goodness of Fit Metric for Multidimensional Outcomes

Tommy Jones* Mark J. Meyer†

2023-12-20

## 1 Introduction

The coefficient of determination—$R^2$—is a popular goodness of fit metric for linear models. It has appealing properties such as a lower bound of zero, an upper bound of one, and interpretable as the proportion of variance in the outcome ~~accounted for by the model~~. $R^2$'s appeal is so strong that nearly all statistical software reports $R^2$ by default when fitting linear models.

Several other pseudo $R^2$ measures have been developed for other use cases, such as Cox and Snell's $R^2$ (Cox and Snell 1989) or McFadden's $R^2$ (McFadden, Tye, and Train 1977). To our knowledge our research is the first time anyone has proposed a variation of $R^2$ for models predicting an outcome in multiple dimensions--where each $\boldsymbol{y}_i$ is a vector. Multidimensional outcomes occur in settings such as modeling simultaneous equations or multivariate distributions. Our $R^2$ ~~relies on~~ a geometric interpretation of the standard definition of $R^2$.

In the unidimensional case, our $R^2$ calculation yields the same result as the traditional $R^2$, with the same properties. In the multidimensional case, there is no lower bound of zero but we argue that this does not negatively affect interpretation. This $R^2$ calculation ~~in an~~ R package (R. C. Team 2013), *mvrsquared*, available on CRAN. (T. C. Team n.d.) In this paper, we ~~introduce the calculation~~, apply it to four multidimensional use cases, and describe use of the *mvrsquared* package.

## 2 A Geometric Interpretation of $R^2$

Let $SSE$ denote the sum of squared errors and $SST$ be the total sum of squares. The standard, model free, definition of $R^2$ is then

*Dept. of Computational and Data Sciences, George Mason University
†Dept. of Mathematics and Statistics, Georgetown University

$$R^2 \equiv 1 - \frac{SSE}{SST} = 1 - \frac{\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}. \tag{1}$$

Each $y_i$ is an observed outcome, $\hat{y}_i$ is the model-based prediction for that outcome, and $\bar{y}$ is the mean outcome across all observations.

$SSE$ and $SST$ may be viewed as a sum of squared Euclidean distances in $\mathbb{R}_1$. Letting $d(p, q) = \sqrt{(p-q)^2}$ for scalar quantities $p$ and $q$, Equation (1) can be re-expressed as

$$R^2 \equiv \frac{\sum_{i=1}^{n} d(y_i, \hat{y})^2}{\sum_{i=1}^{n} d(y_i, \bar{y})^2}. \tag{2}$$

Generalizing to $M$ dimensions, the Euclidean distance becomes $d(\boldsymbol{p}, \boldsymbol{q}) = \sqrt{\sum_{j=1}^{M} (p_j - q_j)^2}$ where $p_j$ and $q_j$ are the $j$th elements of the vectors $\boldsymbol{p}$ and $\boldsymbol{q}$, respectively. Equation (2) is straight-forward to generalize to the $M$ dimensional case. Let $\boldsymbol{y}_i$ and $\hat{\boldsymbol{y}}_i$ be the $M$-dimensional vectors of observed outcomes and predictions for response $i$. We may rewrite Equation (2) as

$$R^2 \equiv= 1 - \frac{\sum_{i=1}^{n} d(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i)^2}{\sum_{i=1}^{n} d(\boldsymbol{y}_i, \bar{\boldsymbol{y}})^2} \tag{3}$$

where $\bar{\boldsymbol{y}}$ is the $M \times 1$ average vector, averaging across all responses.

Figure 1 visualizes the geometric interpretation of $R^2$ for outcomes in $\mathbb{R}_2$. The left image represents $SST$: the red dots are data points ($\boldsymbol{y}_i$); the black dot is the vector of means ($\bar{\boldsymbol{y}}$); the line segments represent the Euclidean distance from each $\boldsymbol{y}_i$ to $\bar{\boldsymbol{y}}$. $SST$ is obtained by squaring the length of each line segment and then adding the squared segments together. The right image represents $SSE$: the blue dots are the fitted values ($\hat{\boldsymbol{y}}_i$); the line segments represent the Euclidean distance from each $\hat{\boldsymbol{y}}_i$ to its corresponding $\boldsymbol{y}_i$. $SSE$ is obtained by squaring the length of each line segment and then adding the squared segments together.

One may also compute the partial $R^2$ as in the traditional case using the formula

$$R_k^2 = 1 - \frac{SSE_{partial}}{SSE_{full}}, \tag{4}$$

Where $SSE_{partial}$ is the sum of squared errors of a model excluding the $k$-th predictor variable.

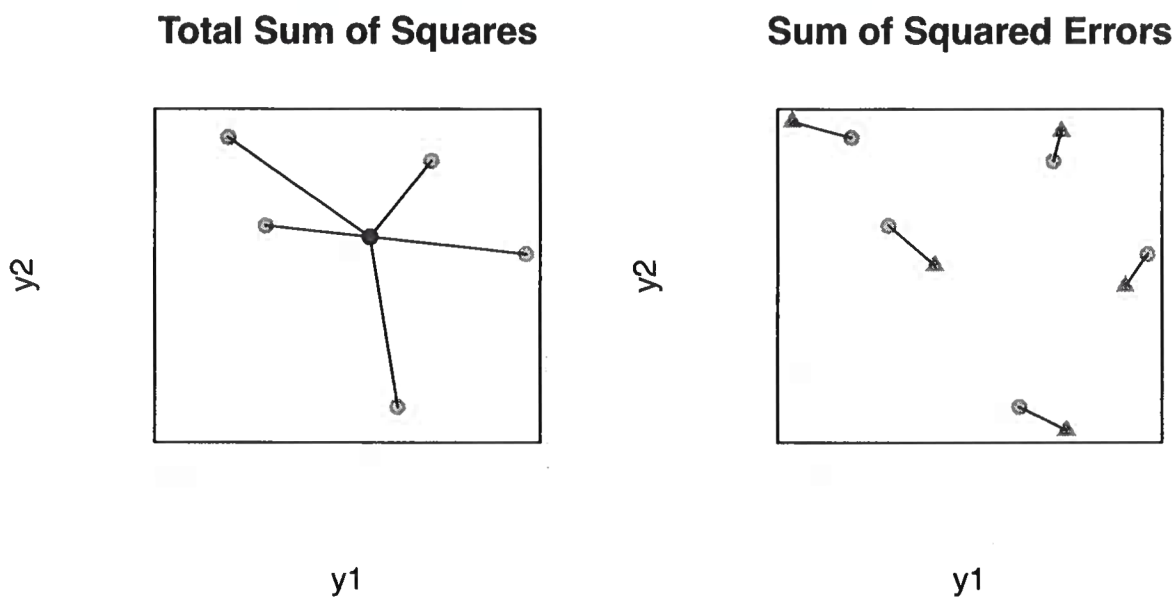


Figure 1: Visualizing the geometric interpretation of R-squared. Sum up the squared length of each line segment for the total (left) or residual (right) sums of squares. This figure corresponds to an R-squared of 0.87

## 2.1 Properties of $R^2$ for Multidimensional Outcomes

$R^2$ for multidimensional outcomes is maximized at one—representing perfect predictions. This can be observed by setting $\boldsymbol{y}_i = \hat{\boldsymbol{y}}_i \; \forall \; i$. It does not have a lower bound, as can be the case in certain circumstances with traditional $R^2$ described by (Barten 1987) and (Cameron and Windmeijer 1997). Its interpretation remains straightforward even without a lower bound. When $R^2 = 0$, then $SSE = SST$ and the model is no better than predicting the mean outcome for each observation. ~~Clearly~~, negative values of $R^2$ mean the model is even worse than predicting the mean outcome for every observation.

The $R^2$ given in (3) is sensitive to the scale of the dimensions of the $\boldsymbol{Y}$, the $n \times M$ matrix of responses. This is a common property of Euclidean distance, exacerbated by squaring. Possible mitigating steps include standardizing the columns of $\boldsymbol{Y}$ before modeling or standardizing both the columns of $\boldsymbol{Y}$ and $\hat{\boldsymbol{Y}}$ after modeling. Results may differ between $R^2$ values calculated on standardized vs non-standardized responses, although we do not explore this in this manuscript.

# 3 Applications

We demonstrate the use of $R^2$ ~~on~~ for multidimensional outcomes on four use cases: two multivariate regression models with varying values of $M$ and scalar covariates, a function-on-function regression model, and a probabilistic topic modeling. In the regression contexts, we employ the multivariate $R^2$ to conduct a simplified model selection for the purpose of illustration.

## 3.1 Bivariate Response

(**Rudorfer1982?**) discusses a small dataset of 17 participants who experienced an overdose of the drug amitriptyline, a tricyclic antidepressant (TCAD) used to treat depression in adults. Since patients may be on multiple antidepressants, the bivariate outcomes of interest are the total TCAD plasma level in the blood and the amount of amitriptyline present in TCAD plasma levels. Potential covariates include sex, amount of antidepressants taken at time of overdose, the PR wave measurement from an electrocardiogram (ECG), diastolic blood pressure, and the QRS wave measurement from an ECG. The primary covariate of interest is the amount of antidepressants taken, so our model selection begins with that variable. We then consider two covariate, three covariate, four covariate, and, if necessary, five covariate models. The base model for the three covariate and four covariate models is the model with the highest percent change from the previous stage's best model. Thus the two covariate model that provided the largest increase in $R^2$ from the model with amount alone is the base model for the three covariate model, and so forth. These results are in Table ??.

Discuss results here. mm add

## 3.2 Response with $M = 4$

In their text on Multivariate Analysis, (**JohnsonWichern2007?**) describe a dataset on paper manufacturing. There are four outcomes of interest: break length, elastic modulus, stress at failure, and burst strength. Covariates include the properties of the pulp fibers: arithmetic fiber length, long fiber fraction, fine fiber fraction, and zero span tensile. We consider a similar model selection approach to the bivariate case, but because there is no primary covariate of interest, we first fit all single covariate models, selecting the one with the highest $R^2$ as the base model for the two covariate models, and so on. Table ?? contains the results of our model selection process.

Discuss results here. mm add

Table 1: R-squared for Function on Function Regression

| Name | R-Squared |
|---|---|
| FFR | 0.871 |
| HFP | 0.453 |
| FH | 0.918 |

### 3.3 Function-on-function Regression

Function-on-function regression (FFR) is a regression framework where both the response and the predictor (or predictors) are functions of time. The time domains need not necessarily align, although for a subclass of FFR models there is a requirement that measurements in predictor occur before or at least concurrently with the response. This subclass of models is referred to as a Historical Functional Linear Model (HFLM). FFR models can include predictors with unrestricted time domains. The relationship between such predictors and the response is described by a surface. When a restriction is imposed for the historical relationship in an HFLM, the surface is truncated to at most the upper triangular region—although other forms of constraint are possible. For more on FFRs and HFLMs, we direct readers to (**Morris2015?**) and (**Meyer2023?**), respectively.

For this illustration, we consider data from a linguistical study ~~on~~ movement of the bottom lip when a participant says the word "bob.'' The data was modeled using an HFLM by (**Malfait2003?**) in their seminal work on the modeling class. For our analysis, the outcome of interest is the position of the lip with possible functional covariates including the acceleration of the lip and an electromyography (EMG) taken during the experiment. Acceleration as a covariate may have a bidirectional relationship in time with position, thus we treat this as an unrestricted functional covariate. The electrical signals measured by the EMG, however, occur before or at most concurrently with the position. Thus, we model this covariate using a historical effect. Table 1 contains the results from this illustration.

Discuss results here.

### 3.4 Probabilistic Topic Modeling

Topic models predict the frequencies of word occurrences across a corpus of documents using latent variables called "topics". Arguably the most famous such model is Latent Dirichlet Allocation (LDA) (Blei, Ng, and Jordan 2003). LDA uses a Bayesian probability model to fit topics. In a probabilistic topic modeling framework, $\boldsymbol{Y}$ is a matrix of integers whose $i, j$ entries give the count of word $j$ in document $i$. Under the model, the $i$-th prediction is given by $\hat{\boldsymbol{y}}_i = n_i \boldsymbol{\theta}_i \cdot \boldsymbol{B}$. Where $\boldsymbol{B}$ is a matrix of latent topic distributions and $\boldsymbol{\theta}_i$

Table 2: $R^2$ for two SBIR topic models.

| Model size | R-sq. | Topic | Prevalence | Part R-sq. | Top 3 words |
|---|---|---|---|---|---|
| 100 | 0.153 | 5 | 2.01 | 0.009 | project, broader, research |
| | | 100 | 0.51 | 0.000 | fire, system, weapon |
| 150 | 0.179 | 101 | 1.83 | 0.009 | project, broader, research |
| | | 137 | 0.57 | 0.000 | tools, complex, lack |

a vector of topic frequencies in the $i$-th document.

We model 100 and 150 scientific topics from the abstracts of 20,766 of Small Business Innovation Research (SBIR) grants awarded between 2020 and 2022. After applying common text pre-processing steps—removing stopwords, removing very infrequent words, and stripping non-alphabetical characters—the data set contains 21,397 unique words and 2,975,409 word occurrences. We use the `tidytext` (Silge and Robinson 2016) and the `tidyverse` (Wickham et al. 2019) packages for data pre-processing. Modeling is done with the `tidylda` package (T. Jones 2020b)—using `tidylda`'s default values for the LDA priors. `tidylda` calculates $R^2$ by calling `mvrsquared`.

The final results are $R^2 = 0.153$ and $R^2 = 0.179$ for the 100 and 150-topic models, respectively, as seen in Table ??. We also calculate the partial $R^2$ for each topic in both models. Table 2 includes the top 5 words in topics with the highest and lowest partial $R^2$, respectively.

Evidence for using $R^2$ for selecting the number of topics to model (i.e., model selection) is mixed. In a second experiment, we explore model selection on a random sample of 1,000 documents from the SBIR corpus. We fit models over a range of 50 to 300 topics—with a step size of 10—using the same specification as above. Figure 2 displays the $R^2$ for each model, plotted against the number of topics. Rather than depecting a clear peak or ~~a monotonically~~ increasing curve, $R^2$ flattens off at about 150 topics. In an earlier work—chapter 4 of (T. W. Jones 2023)—we explore using our $R^2$ calculation for topic modeling in more detail and find ~~similar~~ results on a different corpus.

Also in (T. W. Jones 2023) $R^2$ for multidimensional outcomes applied to topic models is invariant to most properties of the underlying data save one, document length. Shorter documents tend to produce data sets with higher variance in word frequencies across documents. As a result, $R^2$ tends to be lower, owning to the fact that one is modeling noisier data, consistent with the behavior of traditional $R^2$.

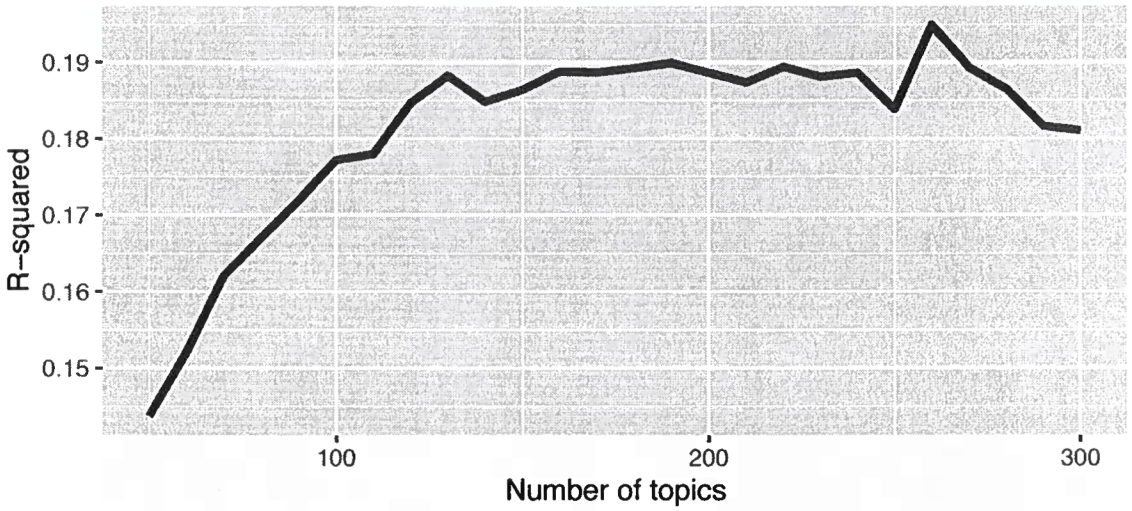


Figure 2: Using R-squared to select the number of topics. Results are mixed. There is neither a peak, nor does the curve monotonically increase.

# 4 The *mvrsquared* Package

We have included our $R^2$ calculation in a package, `mvrsquared`, now available on CRAN. `mvrsquared` contains a single function, `calc_rsquared`, whose arguments are as follows:

- `y` is the true outcome. This must be a numeric vector, numeric matrix, or coercible to a sparse matrix of class `dgCMatrix`.
- `yhat` is the predicted outcome, a vector or matrix, or a list of two matrices whose dot product makes the predicted outcome.
- `ybar` is the mean of `y` and must be a numeric vector for multidimensional outcomes or a scaler for a traditional $R^2$ calculation.
- `return_ss_only` is a logical and controls whether the function returns $R^2$ or $SSE$ and $SST$. Details are described below.
- `threads` is the number of parallel threads the user wants to use for scaling to large data sets.

A simple example of using `mvrsquared` to calculate $R^2$ for univariate data is below. We use univariate data and the dataset `mtcars` for familiarity.

```
library(mvrsquared)

data(mtcars)

# fit a linear model
f <- lm(mpg ~ cyl + disp + hp + wt, data = mtcars)

# extract r-squared for comparison
f_summary <- summary(f)

r2_lm <- f_summary$r.squared

r2_lm
```

```
## [1] 0.8486348
```

```
# calculate univariate r-squared using mvrsquared
r2_mv <- calc_rsquared(y = mtcars$mpg, yhat = f$fitted.values)

r2_mv
```

```
## [1] 0.8486348
```

```
# just to be 100% sure...
r2_lm == r2_mv
```

```
## [1] TRUE
```

When a user wishes to calculate $R^2$, they may either pass `yhat` as matrix (or single vector in the univariate case) of predictions, or they may pass a list with two matrices: `x`, data for the predictor variables, and `w` the linear weights or coefficients of the model. Under the hood, `mvrsquared` will multiply `x` and `w` to obtain `yhat`. (Note that using this method is only valid if a user's model is a linear one.)

```
x <- cbind(1, f$model[, -1]) # note, you have to add 1's for the intercept and
# I'm removing the first column of f$model as it
# is the outcome we are predicting

x <- as.matrix(x) # x needs to be a matrix, not a data.frame or tibble

w <- matrix(f$coefficients, ncol = 1) # w also has to be a matrix

# this calculates yhat as the dot product x %*% w
r2_mv2 <- calc_rsquared(y = mtcars$mpg,
                        yhat = list(x = x,
                                    w = w))

r2_mv2
```

```
## [1] 0.8486348
```

Calculating R-squared this way does lead to a tiny difference in calculation due to numeric precision.

```
r2_mv2 == r2_lm
```

```
## [1] FALSE
```

However, the difference is ~~tiny~~ quite small. Below demonstrates that they are the same up to 14 decimal places in this example.

```
round(r2_mv2, 14) == round(r2_lm, 14)
```

```
## [1] TRUE
```

`mvrsquared` uses the `RcppThread` package for parallelism. [cite] Users tune this behavior with the `threads` argument. In the case where a user has large data—but not so large that it needs to be distributed across machines—setting `threads` to a value greater than one speeds up the calculation. However, in the scenario where the data may be too large to fit one one machine, users can chop up the calculation themselves for

cluster computing, set `return_ss_only` equal to `TRUE`, and then re-combine the results into $R^2$ using $1 - \frac{SSE}{SST}$ and obtaining $SSE$ and $SST$ by summing the intermediate results returned by each node. However if one goes this route, they must pre-calculate `ybar` and pass it to the function to be used in each node. If the user does not, SST will be calculated based on means of each batch independently and the resulting r-squared will be incorrect. Below is a trivial example for illustration using `lapply` instead of proper cluster computing.

```
batch_size <- 10

batches <- lapply(X = seq(1, nrow(mtcars), by = batch_size),
                  FUN = function(b){

                    # rows to select on
                    rows <- b:min(b + batch_size - 1, nrow(mtcars))

                    # rows of the dtm
                    y_batch <- mtcars$mpg[rows]

                    # rows of theta multiplied by document length
                    x_batch <- x[rows, ]

                    list(y = y_batch,
                         x = x_batch)
                  })


# calculate ybar for the data
# in this case, lazily doing colMeans, but you could divide this problem up too
ybar <- mean(mtcars$mpg)

# MAP: calculate sums of squares
ss <- lapply(X = batches,
             FUN = function(batch){
               calc_rsquared(y = batch$y,
```

```
                        yhat = list(x = batch$x, w = w),
                        ybar = ybar,
                        return_ss_only = TRUE)
            })


# REDUCE: get SST and SSE by summation
ss <- do.call(rbind, ss)

ss <- colSums(ss)

r2_mapreduce <- 1 - ss["sse"] / ss["sst"]

# should be the same as above
r2_mapreduce
```

```
##       sse
## 0.8486348
```

# 5 Discussion

Viewing $R^2$ as a ratio of distances motivates a new direction for extended definitions of $R^2$. Researchers may wish to explore $R^2$ calculations based other distance measures more appropriate for other settings. For example, if one's model estimates probabilities perhaps Hellinger distance (Hellinger 1909) would be more appropriate. If scale between outcome variables is of concern, maybe an $R^2$ leveraging Mahalanobis distance (Mahalanobis 1936) has advantage over variable normalization as discussed above.

We have shown that $R^2$ can be calculated for multivariate outcomes from a geometric interpretation of (1). Since we use the standard definition of $R^2$, this approach does not alter existing issues motivating alternate definitions of $R^2$ for, e.g., Bayesian models or nonlinear models. It is worth noting that scale differences in outcome variables can swamp (*TJ: meeaning?*) the calculation. We leave an exploration of remediation strategies to future work. We have implemented the $R^2$ metric in (3) in a package for the R programming language called `mvrsquared` (T. Jones 2020a).

Our multivariate $R^2$ can be used in a number of subfields, some with currently limited model selection tools. In functional regression, various authors have proposed $R^2$-like metrics, see for example approaches taken by (**Meyer2015?**), and references therein. However, there is no generally agreed upon form for the functional $R^2$. Our approach adds to this literature with an alternative that has the added benefit of simplifying to the univariate $R^2$ when $M = 1$. (*TJ: add discussion of lack of model selection tools in topic models*)