The Journal of Open Source Software

dySEM: An *R* Package for Dyadic Structural Equation Modeling with Latent Variables

John Kitchener Sakaluk [1] and **Omar J. Camanto** [1]

1 Department of Psychology, Western University, Canada

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Samuel Forbes

Reviewers:

- @smasongarrison
- @keahtandon

Submitted: 06 January 2026
Published: unpublished

DRAFT

Summary

dySEM is an *R* package (R Core Team, 2024) that was created to make it easier for users to deploy the widely popular lavaan package (Rosseel, 2012) to fit structural equation models (SEMs) using latent variables to data sets with dependent observations (Little, 2013) collected from interdependent dyads (e.g., romantic partners, pairs of friends, parents, a parent and a child, etc.) (Kenny, Kashy, & Cook, 2006). Initially, the package was exclusively designed to facilitate the testing of dyadic invariance (Sakaluk, Fisher, & Kilshaw, 2021)–the psychometric equivalence of survey responses across members of dyads. However, the package has since been expanded to include functions that streamline the process of specifying, fitting, and reporting on a variety of models for dyadic data that are amenable to specification in SEM with latent variables (e.g., Kim & Kim, 2022; Sakaluk et al., 2025).

Currently, dySEM faciltates the deployment of several kinds of dyadic SEMs, including:

- "uni-construct" dyadic models: models in which both members of the dyad are measured on the same, singular construct (e.g., both partners complete a measure of relationship satisfaction)(Sakaluk, 2021; Sakaluk et al., 2025);
- "bi-construct" dyadic models: models in which each member of the dyad is measured on two different constructs, with one being used to predict the other (e.g., the popular Actor-Partner Interdependence Model [APIM], and the less popular Common Fate Model [CFM])(Kenny, 1996, 2018; Kenny et al., 2006; Kim & Kim, 2022; Ledermann & Kenny, 2012)
- "multi-construct" dyadic models: models in which each member of the dyad is measured on multiple constructs, such as when both partners are asked to complete a "multidimensional" survey measure (e.g., a measure of relationship quality with subscales for satisfaction, commitment, intimacy, etc.)

Statement of Need

Models of dyadic data must simultaneously accomplish two analytic goals:

1. Mitigating the impact of non-independent observations on inference when using data collected from members of dyads (which otherwise would inflate rates of false-positive effects)(McCoach & Adelson, 2010); and
2. Somehow representing the theoretical notion of interdependence or "togetherness" that is central to dyadic research (Kenny, 1996; Rusbult & Van Lange, 2003)

Though infrequently adopted (Ledermann & Kenny, 2017; Sakaluk et al., 2025), the combination of (a) the SEM framework with (b) the use of latent variables can provide researchers a powerful analytic framework to accomplish both of these goals, while simultaneously mitigating the biasing (Cole & Preacher, 2014) and Type-I/false-positive boosting impact of measurement error (Westfall & Yarkoni, 2016) that is often present in survey data , and which is particularly pronounced under analytic conditions common to dyadic data analysis (Sakaluk et al., 2025).

But whereas SEM software–like `lavaan` (Rosseel, 2012)–can be used to fit dyadic SEMs with latent variables, the process of specifying, fitting, interpreting, and reporting on such models can be arduous and error-prone, particularly for researchers who are new to SEM or to dyadic data analysis. We see these as considerable barriers to adoption of dyadic SEM, and believe that open-source software can help to lower these barriers (Sakaluk et al., 2025).

`dySEM` is available both on GitHub (https://github.com/jsakaluk/dySEM, https://jsakaluk.github.io/dySE and CRAN (https://cran.r-project.org/web/packages/dySEM/index.html), and was created to simplify the process of using dyadic SEM, making it easier for researchers to leverage the power of dyadic SEMs with latent variables in their own work. Indeed, `dySEM` can typically take an analytic workflow that would otherwise require dozens–if not hundreds–of lines of syntax to be manually written, and reduce it to but a handful or two of lines of (more readable) syntax. For example, the `lavaan` script required for a user to fit a fully distinguishable latent APIM (i.e., no dyadic equality constraints made on any portion of the measurement or structural portions of the model; for a more detailed description, see Kim & Kim (2022))–for hypothetical dyadic data from couples on relationship satisfaction and commitment–would look something like:

```
apim.script <- '

#Measurement Model

#Loadings
Sat1=~NA*sat.g.1_1+sat.g.1_2+sat.g.1_3+sat.g.1_4+sat.g.1_5
Sat2=~NA*sat.g.2_1+sat.g.2_2+sat.g.2_3+sat.g.2_4+sat.g.2_5

Com1=~NA*com.1_1+com.1_2+com.1_3+com.1_4+com.1_5
Com2=~NA*com.2_1+com.2_2+com.2_3+com.2_4+com.2_5

#Residual Variances
sat.g.1_1 ~~ sat.g.1_1
sat.g.1_2 ~~ sat.g.1_2
sat.g.1_3 ~~ sat.g.1_3
sat.g.1_4 ~~ sat.g.1_4
sat.g.1_5 ~~ sat.g.1_5

sat.g.2_1 ~~ sat.g.2_1
sat.g.2_2 ~~ sat.g.2_2
sat.g.2_3 ~~ sat.g.2_3
sat.g.2_4 ~~ sat.g.2_4
sat.g.2_5 ~~ sat.g.2_5

com.1_1 ~~ com.1_1
com.1_2 ~~ com.1_2
com.1_3 ~~ com.1_3
com.1_4 ~~ com.1_4
com.1_5 ~~ com.1_5

com.2_1 ~~ com.2_1
com.2_2 ~~ com.2_2
com.2_3 ~~ com.2_3
com.2_4 ~~ com.2_4
com.2_5 ~~ com.2_5
```

```
#Residual Covariances
sat.g.1_1 ~~ sat.g.2_1
sat.g.1_2 ~~ sat.g.2_2
sat.g.1_3 ~~ sat.g.2_3
sat.g.1_4 ~~ sat.g.2_4
sat.g.1_5 ~~ sat.g.2_5

com.1_1 ~~ com.2_1
com.1_2 ~~ com.2_2
com.1_3 ~~ com.2_3
com.1_4 ~~ com.2_4
com.1_5 ~~ com.2_5

#Structural Model

#Latent (Co)Variances
Sat1 ~~ 1*Sat1
Sat2 ~~ 1*Sat2
Sat1 ~~ Sat2

Com1 ~~ 1*Com1
Com2 ~~ 1*Com2
Com1 ~~ Com2

#Latent Actor Effects
Com1 ~ a1*Sat1
Com2 ~ a2*Sat2

#Latent Partner Effects
Com1 ~ p1*Sat2
Com2 ~ p2*Sat1
'
```

To generate the very same script with dySEM:

```
scriptAPIM(dvn, lvxname = "Sat", lvyname = "Com",
                                constr_dy_x_meas = "none",
                                constr_dy_y_meas = "none",
                                constr_dy_x_struct = "none",
                                constr_dy_y_struct = "none",
                                constr_dy_xy_struct = "none")
```

State of the Field

There are several other *R*-based tools for dyadic data analysis available for free use, owing to a research culture that promotes the generous provision of methodological and analytic support (e.g., Campbell & Kashy, 2002; Ledermann & Kenny, 2015; Stas, Kenny, Mayer, & Loeys, 2018). dySEM, however, is distinctive in both its scope, licensing (and underlying values), and analytic robustness. A comparison of dySEM versus competitor offerings for dyadic SEM with latent variables and observed variable path analysis is available below in **??**. At a high level, dySEM is distinguished by being fully open-source (i.e., all source code is available to interested users, and can be modified and built upon by others following a GNU General Public License) and supported by a large number of (transparent) unit-tests, while offering

The Journal of Open Source Software

convenient functionality for scripting, calculating, and outputting for a large variety of dyadic SEM models (of which, it is further distinguished by offering the greatest amount of support for [and variety of] latent variable models). And when compared to all alternatives, `dySEM` appears to be the only offering for which unit testing for quality-control is extensive (and growing) and made transparent to the user. We therefore immodestly believe `dySEM` is—and will continue to be—among the more competitive software solutions for dyadic SEM.

Table 1: Comparison of dyadic SEM software.

Software	Free	OS Code	OS Tests	Active	Scripting	Documentation
dySEM (*R* Package: GitHub/CRAN)	Yes	Yes	Yes	Yes	13 distinct uni-, bi-, and multi-construct dyadic SEMs	Extensive
"DyadR" Shiny Apps (GUI-Based)	Yes	No	No	Yes	6 models (only 1 latent)	Extensive
srm (*R* Package: GitHub/CRAN)	Yes	Yes	No	No	1 model (non-latent)	Minimal
lavaan.srm (*R* Package: GitHub)	Yes	Yes	No	Yes	1 model (non-latent)	Moderate
lavaan (*R* Package: GitHub/CRAN)	Yes	Yes	No	Yes	None	Extensive
Proprietary software (MPlus, AMOS, SAS, etc.,)	No	No	No	Mixed	None	Minimal

Software Design

Design Philosophy

`dySEM` has been designed for users to follow a generalized 4-step workflow:

1. *scrape* variable information from a data frame of dyadic data
2. *script* the dyadic SEM of interest with one of `dySEM`'s "scripter" " functions
3. *fit* the scripted model with `lavaan`
4. Use `dySEM` "outputter" functions to generate reproducible tabular and graphical summaries of the fitted model

By building on `lavaan` (Rosseel, 2012), `dySEM` leverages a widely used, open-source SEM package (and its companion packages) with a large and active user base, and which is under ongoing development. `dySEM`'s capacities should therefore be able to grow alongside `lavaan`'s.

With `dySEM`, we strive to promote an inclusive data analytic ecosystem. The added demand and complexity of implementing certain dyadic models or calculating certain "corrections" (Olsen & Kenny, 2006) may be one (among many) causes of the under-representation of certain types of couples in research (McGorray, Emery, Garr-Schultz, & Finkel, 2023). To that end, all scripting functionality in `dySEM` has been designed to default to an "indistinguishable" model (i.e., one amenable to analyzing gender- and sexuality-diverse couples)–the user must deliberately over-ride these defaults, putting the burden of proof on them to defend the choice of a distinguishable model and their distinguishing feature (Kenny et al., 2006)–and offers helper functionality to navigate additional indices that are requested when fitting such models (Olsen & Kenny, 2006).

Functionality

`dySEM` requires dyadic data to be available in a data frame that is in "dyad" or "wide" form (i.e., one row per dyad) (Kenny et al., 2006), and with variable names that follow a discernible pattern. `dySEM` "scrapes" this information about variables into a list, which serves as a key input of its scripter functions. For more information, see our tutorial on naming conventions, and documentation for `scrapeVarCross()`.

The largest source of value in `dySEM`'s code-base are its scripters (i.e., functions beginning with the term "script"), which generate character objects of syntax for dyadic SEMs that can be immediately passed to `lavaan` for model fitting, with whatever optionality (e.g., estimator selection, missing data treatment) the user desires. These scripters generally have optionality for:

- what method of scale-setting one uses (i.e., "fixed factor"/standardized latent variance or "marker variable"/constraining a factor loading to 1)(Little, 2013)
- what kinds of dyadic equality constraints one wants to impose on the measurement and/or structural portions of the model (e.g, for dyadic invariance testing; to facilitate dyadic comparisons of structural parameters)(Sakaluk, Fisher, et al., 2021)
- whether to include `lavaan` syntax to compute "boutique" estimates and tests (e.g., the *k* parameter of dyadic patterns for an APIM) (Kenny & Ledermann, 2010), and
- whether to write and export .txt file of the generated script (e.g, to post on the Open Science Framework)

Some examples (with varied use of these arguments):

```
#Example 1: A fully invariant APIM with actor effects constrained
# across partners, partner effects freely estimated

apim.script <-  scriptAPIM(dvnxy, lvxname = "Sat", lvyname = "Com",
constr_dy_x_meas = c("loadings", "intercepts", "residuals"),
constr_dy_y_meas = c("loadings", "intercepts", "residuals"),
constr_dy_xy_struct = c("actors"), est_k = TRUE,
scaleset = "FF")

apim.mod <- lavaan::cfa(apim.script, data = commitmentQ)

#Example 2: A correlated dyadic factors model that is
# "residually invariant" (i.e., constraints on
# the pattern, loadings, intercepts, and residuals across dyad members,
#but none for the parameters in the structural portion of the model)
# and which writes ResidualInvariance.txt to working directory

sat.resid.script <- scriptCor(dvn, lvname = "Sat",
                              constr_dy_meas = c("loadings",
                              "intercepts",
                              "residuals"),
                              constr_dy_struct = "none",
                              writeTo = ".",
                              fileName = "ResidualInvariance")
```

```r
sat.resid.mod <- lavaan::cfa(sat.resid.script, data = commitmentM,
estimator = "mlr", missing = "ml")
```

dySEM also provides assistance with easy, reproducible reporting from these models. Outputter functions (beginning with "output") can be used to:

- generate tables of parameter estimates from either/both the measurement portion and/or structural portion of a model
- generate path diagrams via the semPlot package (Epskamp, 2015) for visualizing model structure and parameter estimates
- generate tables of model comparisons (e.g., in dyadic invariance testing)
- generate tables of "boutique" estimates and tests (e.g., Langrange multiplier tests for identifying item/parameter-sources of noninvariance; correlations among latent variables), and
- compute "boutique" values (e.g., alternative metrics of reliability; effect sizes of dyadic noninvariance)

When tables are created, they can be kept in either data frame form (e.g., to supply visualizations), or exported as .rtf, while path diagrams are exported as .png.

For example:

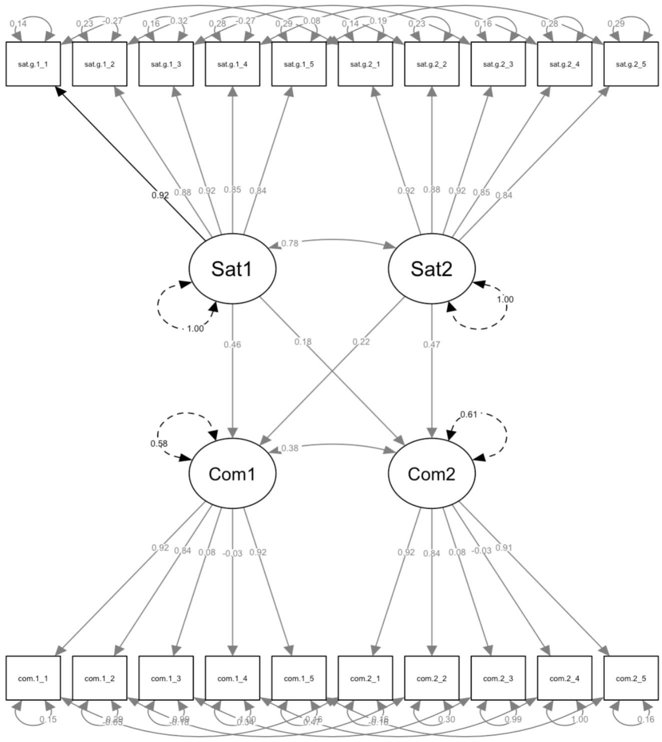
```r
#Example 1: Export a table of measurement parameter estimates
# named APIM_Measurement_Table, to working directory.
# Table is generated by gt::gt()
outputParamTab(dvnxy, model = "apim", gtTab = TRUE ,
               apim.mod, tabletype = "measurement",
               writeTo = ".", fileName = "APIM_Measurement_Table")
```

Latent Factor	Indicator	Loading	SE	Z	p-value	Std. Loading	Intercept
Sat1	sat.g.1_1	1.976	0.129	15.290	< .001	0.925	6.609
Sat1	sat.g.1_2	1.858	0.137	13.612	< .001	0.878	6.591
Sat1	sat.g.1_3	1.980	0.130	15.175	< .001	0.917	6.391
Sat1	sat.g.1_4	1.775	0.133	13.387	< .001	0.851	6.673
Sat1	sat.g.1_5	1.891	0.145	13.075	< .001	0.844	6.445
Sat2	sat.g.2_1	1.976	0.129	15.290	< .001	0.925	6.918
Sat2	sat.g.2_2	1.858	0.137	13.612	< .001	0.878	6.927
Sat2	sat.g.2_3	1.980	0.130	15.175	< .001	0.917	6.727
Sat2	sat.g.2_4	1.775	0.133	13.387	< .001	0.851	7.155
Sat2	sat.g.2_5	1.891	0.145	13.075	< .001	0.844	6.864
Com1	com.1_1	1.459	0.092	15.875	< .001	0.921	7.527
Com1	com.1_2	1.348	0.094	14.374	< .001	0.842	7.236
Com1	com.1_3	0.194	0.177	1.096	0.273	0.083	4.809
Com1	com.1_4	-0.076	0.176	-0.430	0.667	-0.032	4.300
Com1	com.1_5	1.440	0.090	15.933	< .001	0.918	7.282
Com2	com.2_1	1.459	0.092	15.875	< .001	0.916	7.327
Com2	com.2_2	1.348	0.094	14.374	< .001	0.835	7.345
Com2	com.2_3	0.194	0.177	1.096	0.273	0.080	5.118
Com2	com.2_4	-0.076	0.176	-0.430	0.667	-0.031	4.136
Com2	com.2_5	1.440	0.090	15.933	< .001	0.914	7.191

: outputParamTab( ).

The Journal of Open Source Software

```
#Example 2: Export a path diagram named
# APIM Diagram.png to working directory
# named APIM_Measurement_Table, to working directory
outputParamFig(apim.mod, figtype = "standardized",
               writeTo = ".", fileName = "APIM Diagram")
```



: outputParamFig( ).

```
#Example 3: Generate a table of Langrange Multiplier tests
# for a dyadic invariance model. Do not filter for
# significance, and output as a gt::gt() table.
outputConstraintTab(sat.resids.mod, filterSig = FALSE, gtTab = TRUE)
```

param1	constraint	param2	chi2	df	pvalue	sig
Satf =~ sat.g1_f	==	Satm =~ sat.g1_m	1.131	1	0.288	NA
Satf =~ sat.g2_f	==	Satm =~ sat.g2_m	0.633	1	0.426	NA
Satf =~ sat.g3_f	==	Satm =~ sat.g3_m	0.060	1	0.806	NA
Satf =~ sat.g4_f	==	Satm =~ sat.g4_m	1.839	1	0.175	NA
Satf =~ sat.g5_f	==	Satm =~ sat.g5_m	3.603	1	0.058	NA
sat.g1_f ~1	==	sat.g1_m ~1	0.057	1	0.812	NA
sat.g2_f ~1	==	sat.g2_m ~1	1.316	1	0.251	NA
sat.g3_f ~1	==	sat.g3_m ~1	0.048	1	0.827	NA
sat.g4_f ~1	==	sat.g4_m ~1	0.103	1	0.748	NA
sat.g5_f ~1	==	sat.g5_m ~1	2.090	1	0.148	NA
sat.g1_f ~~ sat.g1_f	==	sat.g1_m ~~ sat.g1_m	22.977	1	0.000	***
sat.g2_f ~~ sat.g2_f	==	sat.g2_m ~~ sat.g2_m	0.263	1	0.608	NA
sat.g3_f ~~ sat.g3_f	==	sat.g3_m ~~ sat.g3_m	0.317	1	0.573	NA
sat.g4_f ~~ sat.g4_f	==	sat.g4_m ~~ sat.g4_m	2.422	1	0.120	NA
sat.g5_f ~~ sat.g5_f	==	sat.g5_m ~~ sat.g5_m	17.185	1	0.000	***

: `outputConstraintTab()`.

Research Impact

Since its initial release on GitHub in 2021, `dySEM` has been accepted to CRAN (where it has been downloaded more than 4,000 times). `dySEM` has been used in published research, both by members of our research team (e.g., Sakaluk, Quinn-Nilas, et al., 2021) and by other teams in the field (e.g., Girme & Overall, 2025). `dySEM` has also featured in workshops we have delivered at conferences for which dyadic data analysis is an emphasis, and dyadic SEM as a framework, more generally, is enjoying a phase of renewed methodological interest (e.g. Joel, Eastwick, & Khera, 2025; Sakaluk et al., 2025).

AI Usage Disclosure

`dySEM`'s development began in 2019, well before the advent of mainstream AI tools, and the bulk of the package's current design, functionality, and roadmap have been developed without AI assistance. Our development team now uses AI tools (e.g., Copilot autocompletion within RStudio; increasing use of Cursor as an AI-boosted IDE), though primarily to assist with rote and/or repetitious tasks (e.g., to increase testing coverage of various input or output requirements; to increase consistency of documentation across related functions, etc.). A critical mass of new functionality and its more substantive testing remains done "by hand", and this manuscript was written and submitted with limited AI assistance (i.e., table formatting; proper implementation of the Open Journals PDF Generator GitHub action).

Acknowledgements

We wish to thank Drs. William Chopik, Christopher Chartier, Lorne Campbell, and Brent Donnellan for their facilitating the special issue of Personal Relationships that led to `dySEM`'s creation (Chopik, Chartier, Campbell, & Brent Donnellan, 2020). We are also very appreciative for the late Dr. Deborah Kashy's enthusiasm towards `dySEM`'s creation and functionality.

The development of `dySEM` has been supported by three internal grants from Western University:

- Western Knowledge Mobilization and Research-Creation Grant (2024-2025)
- Western Research Mobilization, Creation, & Innovation Grants for SSHRC-related Research (2023-2024)
- Western Knowledge Mobilization Innovation Grants (2021-2022)

References

Campbell, L., & Kashy, D. A. (2002). Estimating actor, partner, and interaction effects for dyadic data using PROC MIXED and HLM: A user-friendly guide. *Personal Relationships*, *9*(3), 327–342. doi:10.1111/1475-6811.00023

Chopik, W. J., Chartier, C. R., Campbell, L., & Brent Donnellan, M. (2020). Relationship science and the credibility revolution: An introduction to the first part of the special issue. *Personal Relationships*, *27*(1), 132–137. doi:10.1111/pere.12312

Cole, D. A., & Preacher, K. J. (2014). Manifest variable path analysis: Potentially serious and misleading consequences due to uncorrected measurement error. *Psychological Methods*, *19*(2), 300–315. doi:10.1037/a0033805

Epskamp, S. (2015). semPlot: Unified visualizations of structural equation models. *Structural Equation Modeling: A Multidisciplinary Journal*, *22*(3), 474–483. doi:10.1080/10705511.2014.937847

Girme, Y. U., & Overall, N. C. (2025). Greater average levels of relatedness need fulfilment across daily and monthly life predict lower attachment insecurities across time. *Journal of Social and Personal Relationships*, *42*(5), 1127–1148. doi:10.1177/02654075231162390

Joel, S., Eastwick, P. W., & Khera, D. (2025). A credibility revolution for relationship science: Where can we step up our game? *Social and Personality Psychology Compass*, *19*(2). doi:10.1111/spc3.70042

Kenny, D. A. (1996). Models of non-independence in dyadic research. *Journal of Social and Personal Relationships*, *13*(2), 279–294. doi:https://doi.org/10.1177/0265407596132007

Kenny, D. A. (2018). Reflections on the actor-partner interdependence model. *Personal Relationships*, *25*(2), 160–170. doi:10.1111/pere.12240

Kenny, D. A., Kashy, D. A., & Cook, W. L. (2006). *Dyadic data analysis*. New York, NY: Guilford Press.

Kenny, D. A., & Ledermann, T. (2010). Detecting, measuring, and testing dyadic patterns in the actor–partner interdependence model. *Journal of Family Psychology*, *24*(3), 359–366. doi:10.1037/a0019651

Kim, H., & Kim, J.-S. (2022). Extending the actor-partner interdependence model to accommodate multivariate dyadic data using latent variables. *Psychological Methods*. doi:10.1037/met0000531

Ledermann, T., & Kenny, D. A. (2012). The common fate model for dyadic data: Variations of a theoretically important but underutilized model. *Journal of Family Psychology*, *26*(1), 140–148. doi:10.1037/a0026624

Ledermann, T., & Kenny, D. A. (2015). A toolbox with programs to restructure and describe dyadic data. *Journal of Social and Personal Relationships*, *32*(8), 997–1011. doi:10.1177/0265407514555273

Ledermann, T., & Kenny, D. A. (2017). Analyzing dyadic data with multilevel modeling versus structural equation modeling: A tale of two methods. *Journal of Family Psychology*, *31*(4), 442–452. doi:10.1037/fam0000290

Little, T. D. (2013). *Longitudinal structural equation modeling*. New York, NY: Guilford Press.

McCoach, D. B., & Adelson, J. L. (2010). Dealing with dependence (part i): Understanding the effects of clustered data. *Gifted Child Quarterly*, *54*(2), 152–155. doi:10.1177/0016986210363076

McGorray, E. L., Emery, L. F., Garr-Schultz, A., & Finkel, E. J. (2023). "Mostly white, heterosexual couples": Examining demographic diversity and reporting practices in relationship science research samples. *Journal of Personality and Social Psychology*, *125*(2), 316–344. doi:10.1037/pspi0000417

Olsen, J. A., & Kenny, D. A. (2006). Structural equation modeling with interchangeable dyads. *Psychological Methods*, *11*(2), 127–141. doi:10.1037/1082-989X.11.2.127

R Core Team. (2024). R: A language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Rosseel, Y. (2012). **Lavaan** : An *r* package for structural equation modeling. *Journal of Statistical Software*, *48*(2). doi:10.18637/jss.v048.i02

Rusbult, C. E., & Van Lange, P. A. M. (2003). Interdependence, interaction, and relationships. *Annual Review of Psychology*, *54*(1), 351–375. doi:10.1146/annurev.psych.54.101601.145059

Sakaluk, J. K. (2021). Introducing the bifactor dyadic (BiDy) model: New possibilities for the analysis of dyads. Presented at the Relationships preconference of the annual meeting of the society for personality and social psychology, Virtual/Online.

Sakaluk, J. K., Fisher, A. N., & Kilshaw, R. E. (2021). Dyadic measurement invariance and its importance for replicability in romantic relationship science. *Personal Relationships*, *28*(1), 190–226. doi:10.1111/pere.12341

Sakaluk, J. K., Joel, S., Quinn-Nilas, C., Camanto, O. J., Pevie, N. W., Tu, E., & Jorgensen-Wells, M. A. (2025). A renewal of dyadic structural equation modeling with latent variables: Clarifications, methodological advantages, and new directions. *Social and Personality Psychology Compass*, *19*, e70045. doi:https://doi.org/10.1111/spc3.70045

Sakaluk, J. K., Quinn-Nilas, C., Fisher, A. N., Leshner, C. E., Huber, E., & Wood, J. R. (2021). Sameness and difference in psychological research on consensually non-monogamous relationships: The need for invariance and equivalence testing. *Archives of Sexual Behavior*, *50*(4), 1341–1365. doi:10.1007/s10508-020-01794-9

Stas, L., Kenny, D. A., Mayer, A., & Loeys, T. (2018). Giving dyadic data analysis away: A user-friendly app for actor–partner interdependence models. *Personal Relationships*, *25*(1), 103–119. doi:10.1111/pere.12230

Westfall, J., & Yarkoni, T. (2016). Statistically controlling for confounding constructs is harder than you think. (U. S. Tran, Ed.)*PLOS ONE*, *11*(3), e0152719. doi:10.1371/journal.pone.0152719