# riskmetric: A Risk-based Workflow to Evaluate the Quality of R Packages

Douglas Kelkhoff[1], Yilong Zhang[2], Eli Miller[3]
Eric Milliman[4], Marly Gotti[4], Juliane Manitz[5], Mark Padgham[6]
[1]Genentech, 1 DNA Way, South San Francisco, CA 94080, USA
[2]Merck & Co., Inc., Kenilworth, NJ, USA
[3]Atorus Research, Madison, WI, USA
[4]Bioegen Inc., 300 Binney St, Cambridge, MA 02142, USA
[5]EMD Serono, 45 Middlesex Tpk, Billerica, MA 01821, USA
[6]rOpenSci & EcoHealth Alliance, 20 Eighth Avenue, Ste. 1200, New York, NY 10018, USA

**Abstract**
Open source tools offer easier access to leading edge methods than ever before, but with that comes new risks of code quality that aren't addressed by familiar regulated industry practices. At the R Validation Hub, we offer the R package `riskmetric` to help bridge that gap. The package provides a platform for quantifying the quality of an R package, supporting risk-based validation of software opening the door for faster and more dynamic incorporation of open source tools into regulatory analysis.

We will give a walkthrough of the riskmetric package, highlighting potential use cases ranging from scientific end users to systems administrators, and discuss how the package contributes one piece of the R Validation Hub's software strategy for advancing the role of open source tools in the biopharmaceutical clinical development process.

**Key Words:** R package, validation, open source, software development, software risk

## 1. Introduction

Validation is a core component of the lifecycle of software tools used for decision making in regulated industries, such as the pharmaceutical industry. Software validation should provide confidence in the installation and user-facing behaviors of a given piece of software. Inevitably, the stringency of this process is governed by the perceived risk of a software component, applying extra caution with tools that may be subject to regulatory review including interactions with patient data and analysis for regulatory decision making. This evaluation comprises the risk assessment of any software component prior to validation.

### 1.1 Validation in the Context of Modern Package Repositories

Within the pharmaceutical industry, pre-configured and centrally administered systems have been the focal point of validation. Such an approach is convenient when analytic

tools are provided as a packaged distribution. However, modern statistical and analytic tools, such as R and its surrounding package ecosystem, are more commonly distributed as modular components or packages. With this added flexibility built into an analytic environment comes the challenge of assessing the risk of an entire cohort of tools, individually validating them for regulated use and evaluating any changes to the system to determine if changes comporomise the validity of results. To aid in this decision making process, our team has developed the `riskmetric` R package, which provides a selection of risk assessments as well as an extensible framework for incorporating future assessments[1]. This work is part of a broader effort as part of the R Validation Hub to provide guidance and tools to support the validation of R environments, with a focus on applications in the pharmaceutical industry[2].

**1.2 Measures of Software Quality**

Measuring software risk is not a well defined problem, and decision making is often governed by interpretation of many heuristics used to best approximate the quality and reliability of software. There are many dimensions to the problem of assessing risk, and each has their own heuristics that might inform a risk decision. Furthermore, some risks may be more permissible than others and might be better handled by the host system where the software is to be installed. Broadly, categories of risk may be divided into:

*1.2.1 Technical Quality*

Within the R language, technical quality is broadly addressed by the built in `R CMD check` utility, assuring that a package meets basic requirements to be appropriately distributed and tests built into the package pass on the given environment.

*1.2.2 Community Adoption*

User adoption can be a good indicator of reliability. High quality open-source tools often attract a larger number of users, offering more attention to potential software bugs and a higher likelihood of long-term maintenance.

*1.2.3 Development Practices*

Adherence to accepted best practices can indicate core competencies of the development team and can provide assurances that the software will maintain a fundamental level of quality through such activities as continuously integrated testing and behavioral guidelines such as a contribution guide and code of conduct. Beyond the R language, guidance and evaluation of open source development practices has drawn the attention of language-agnostic efforts to establish best practices[3,4].

*1.2.4 Security Considerations*

Naturally, any software used in a regulated setting is subject to security scrutiny, and R packages are no exception. Broadly speaking, security considerations are better controlled on a host system than within each individual R package. Nevertheless, there are cursory evaluations that can be done to highlight potentially insecure behaviors and check the use of components that have known vulnerabilities.

*1.2.5 Inter-package Compatibility*

Finally, R packages do not exist in isolation and should be evaluated within the context of their software dependencies and the host system. Individual package behaviors may be affected by the availability of software dependencies, and the risk of those package behaviors may be affected. This is still an active area of development.

## 2. `riskmetric` Package Overview

### 2.1 Package Overview

At a high level, the `riskmetric` package provides an interface to collect package information from a variety of frequent sources of package information. This may include public package repositories, public source code repositories, local source code or locally installed package libraries. This information builds a collection of package metadata which may be helpful for a risk assessment. This metadata is used to derive a number of package metrics – heuristics which may be helpful atomic criteria that can be compared across packages. Finally, these criteria are scored, consolidating metrics into numeric indications of their quality and allowing for aggregate comparison of individual packages or package cohorts.

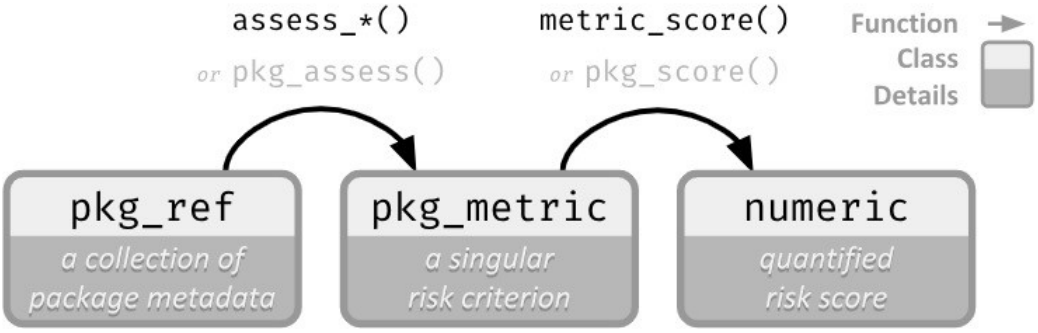


**Figure 1:** riskmetric package data flow overview.

### 2.2 Design Goals

Beyond being a tool for assessing packages, `riskmetric` is designed to be an extensible foundation for customizing a risk assessment workflow, allowing enterprises and regulated industry contributors to tailor a risk assessment to their application. This includes the addition of new metrics, customizing the scoring functions for existing or new metrics and defining an aggregating function to summarize overall package risk. Likewise, these hooks for customization allow for more rapid experimentation and contribution, allowing for an accessible path from user to contributor.

Additionally, a core design goal of `riskmetric` was to handle many of the challenges of metric execution order. As metrics may rely on similar, and occasionally computationally intensive derived attributes of packages, the execution engine was designed such that these relationships need not be predefined before computation, freeing developers of the cognitive burden of managing the collection of metadata.

### 2.3 Example Use

To highlight how `riskmetric` may be used, the following R code is provided to showcase a simplified workflow. Included below is the R code equivalent of the workflow described in Figure 1. Shown below are the package scores calculated by `riskmetric`. As well, each intermediate function call could likewise be use to inspect intermediate heuristics and package data.

```
library(riskmetric)
library(dplyr)
```

```
pkg_ref(c("riskmetric", "utils", "tools")) %>%
  as_tibble() %>%
  pkg_assess() %>%
  pkg_score()

#> # A tibble: 3 × 18
#>   package    version pkg_ref             pkg_score news_current
#>   <chr>      <chr>   <list<pkg_ref>>         <dbl> <pkg_scor>
#> 1 riskmetric 0.1.0   riskmetric<install>     0.399 1
#> 2 utils      4.0.3   utils<install>          0.786 0
#> 3 tools      4.0.3   tools<install>          0.857 0
#> # … with 13 more variables: has_vignettes <pkg_scor>,
#> #   has_bug_reports_url <pkg_scor>, bugs_status <pkg_scor>,
#> #   license <pkg_scor>, export_help <pkg_scor>,
#> #   downloads_1yr <pkg_scor>, has_website <pkg_scor>,
#> #   r_cmd_check <pkg_scor>, remote_checks <pkg_scor>,
#> #   has_maintainer <pkg_scor>, has_news <pkg_scor>,
#> #   has_source_control <pkg_scor>, covr_coverage <pkg_scor>
```

## 3. Future Work

`riskmetric` represents a community effort to make the risk assessment process more reliable, consistent and user friendly. As such, it is under constant development, with new areas of interest continuing to surface.

Currently, an area of interest is the calculation of metrics which are dependent on the context in which they are evaluated. Notably, this includes considerations about the dependency structure of a package library and the interoperability of installed packages. Additional work is ongoing to provide a graphical interface to the riskmetric functionality to improve ease-of-use for non-technical users.

## Acknowledgements

We'd like to thank the R Validation Hub, and its sponsors the R Consortium and the PSI/EFSPI AIMS SIG. We'd also like to recognize the contributions and support from ROpenSci whilst designing the architecture of riskmetric and continuing to provide input for key design decisions.

## References

1. *riskmetric. riskmetric: Metrics to evaluate the risk of r packages*. GitHub. Retrieved September 14, 2021, from https://github.com/pharmaR/riskmetric.
2. R Validation Hub. Retrieved September 14, 2021, from https://www.pharmar.org/.
3. *Open Source Security Foundation (openssf)*. Open Source Security Foundation. (2021, January 14). Retrieved September 14, 2021, from https://openssf.org/.
4. *Open source guides*. Linux Foundation. (2021, April 5). Retrieved September 14, 2021, from https://www.linuxfoundation.org/resources/open-source-guides/.