Building and Maintaining a Bioconductor Package

A practical guide

Ossama Edbali (o.edbali@bham.ac.uk)

What we'll cover

01
Why Bioconductor?
How it differs from CRAN and what it is

02
Package anatomy
DESCRIPTION, NAMESPACE, R/, man/, tests/, vignettes/

03
Architectural decisions
S4, biocViews, the obligation to reuse the ecosystem

04
Documentation and vignettes
roxygen2, man pages, fully executable vignettes

05
BiocCheck
The automated gate between your code and acceptance

06
Submission and peer review
Pre-submission prep and the review dialogue

07
Acceptance and maintenance
Dual-remote git, the release cycle, maintenance best practices

Bioconductor

CRAN	Bioconductor
• General-purpose R packages	• Domain-specific: genomics, proteomics, metabolomics
• Rolling submissions, automated checks	• Formal peer review by a Bioconductor reviewer
• Minimal review (compared to Bioconductor)	• Shared data structures across the ecosystem
• No prescribed domain or data classes	• Two coordinated releases per year (April and October)
• One stable stream of versions	• Devel + Release branches, synchronised with R
• Community support: mailing lists, Stack Overflow	• Dedicated support site + bioc-devel mailing list

Types of packages

Software packages

Provide implementation of algorithms (e.g. statistical analysis), access to resources (e.g. biomaRt, or NCBI) or visualizations (e.g. volcano plots, pathways plots).

Annotation packages

Database-like packages that provide information linking identifiers (e.g., Entrez gene names or Affymetrix probe ids) to other information (e.g., chromosomal location, Gene Ontology category).

Experiment data packages

Provide data sets that are used, often by software packages, to illustrate particular analyses

Workflow packages

Contain vignettes that describe a bioinformatics workflow that involves multiple Bioconductor packages.

Use case: lcmsPlot

An R Bioconductor package for liquid-chromatography mass-spectrometry (LC-MS) metabolomics data visualisation with a ggplot2-style layering API that interoperates with the existing Bioconductor MS stack (MsExperiment, MSnbase, xcms).

With base xcms plotting

R

```
chr <- chromatogram(raw_data,
        mz = c(300, 305),
        rt = c(2500, 2550))
par(mfrow = c(n_row, n_col))
for (i in seq_along(chr)) {
  df <- as.data.frame(chr[[i]])
  plot(df$rtime, df$intensity,
       type = "l", col = group_cols[group])
  abline(v = 2520, lty = 2)
}
```

With lcmsPlot

R

```
lcmsPlot(raw_data, sample_id_column = "sample_name") +
  chromatogram(features = rbind(c(
    mzmin = 300, mzmax = 305,
    rtmin = 2500, rtmax = 2550))) +
  rt_line(intercept = 2520) +
  arrange(group_by = "sample_group") +
  facets(facets = "sample_id", ncol = 3)
```

Package anatomy

```
lcmsPlot/
├── DESCRIPTION
├── NAMESPACE
├── NEWS.md
├── README.md
├── LICENSE
├── R/
│   ├── lcmsPlot-class.R
│   ├── chromatogram.R
│   └── ...
├── man/
│   └── *.Rd
├── tests/
│   └── testthat/
├── vignettes/
│   └── lcmsPlot.Rmd
└── inst/extdata/
```

DESCRIPTION

Metadata: name, version, dependencies, biocViews

R/

One concept per file; export via NAMESPACE

man/

Generated by roxygen2

tests/

testthat is strongly recommended

vignettes/

At least one: must build and run during checks

inst/extdata/

Small example data (< 5 MB total package)

The DESCRIPTION file

DCF

```
Package: lcmsPlot
Type: Package
Title: Comprehensive Liquid Chromatography-Mass Spectrometry (LC-MS) data visualisation
package
Version: 0.99.20
Authors@R: c(...)
Description: lcmsPlot is an R package designed for
    visualising Liquid Chromatography-Mass Spectrometry
    (LC-MS) data with publication-ready high-quality plots.
License: GPL-3
Depends: R (>= 4.4.0)
Imports: methods, rlang, dplyr, tibble, tidyr,
    BiocParallel, MSnbase, xcms, MsExperiment, mzR,
    Spectra, MsBackendMsp, S4Vectors, ggplot2, scales,
    patchwork, DBI, RSQLite
Suggests: knitr, rmarkdown, BiocStyle, openxlsx,
    faahKO, rawrr, testthat (>= 3.0.0)
VignetteBuilder: knitr
biocViews: Metabolomics, MassSpectrometry
URL: https://github.com/computational-metabolomics/lcmsPlot/issues
BugReports: https://github.com/.../lcmsPlot/issues
Config/testthat/edition: 3
```

biocViews and the ecosystem

Tags from a **controlled vocabulary** describing what your package does. They drive the **BiocViews browser**, how users discover your package, and which category it belongs to.

Four parent categories

`Software`	Analysis code (most packages)
`ExperimentData`	Example datasets for other packages
`AnnotationData`	Annotation objects (genomes, pathways)
`Workflow`	End-to-end guided analyses

Re-use, don't reinvent

Bioconductor expects packages to interoperate via shared S4 classes. lcmsPlot doesn't define new MS container classes; it consumes existing ones:

- MsExperiment – MS experiment container
- Msnbase – spectra, chromatograms container
- xcms – LC-MS processing and analysis
- S4Vectors – Bioconductor's base S4 toolkit

Architectural decisions in lcmsPlot

Compose over invent. lcmsPlot accepts 9 upstream classes, delegates data extraction to MS backends such as mzR, and renders with patchwork on top of ggplot2.

Layered S4 design

lcmsPlotClass wraps an lcmsPlotDataContainer. Plot state (options, history, rendered patchwork) stays separate from data state so layers can replay without touching raw MS data.

```
setClass("lcmsPlotClass",
  slots = list(
    options = "list",
    data    = "lcmsPlotDataContainer",
    history = "list",
    plot    = "ANY"))
```

The + operator working similarly to ggplot2

Each lp_* constructor returns a closure. The + method simply applies it. This gives familiarity to users of ggplot2.

```
setMethod(
  f = "+",
  signature = c("lcmsPlotClass", "function"),
  definition = function(e1, e2) e2(e1)
)
```

Nine input types, one entry point

Supports varied input types from the MS Bioconductor ecosystem. These include XCMSnExp, MsExperiment,character (raw sample paths), DBIConnection, and so on.

```
get_metadata <- function(obj, ...) {
  UseMethod("get_metadata")
}
get_metadata.MsExperiment <- ...
get_metadata.XCMSnExp     <- ...
get_metadata.character    <- ...
```

Layer history enables replay

Every lp_* call is recorded in @history before it runs. That's how next_plot() and iterate_plot_batches() move through large datasets without re-specifying the plot.

```
# closure captures args + records history
function(obj, record_history = TRUE) {
  if (record_history)
    obj@history <- c(obj@history,
      list(list(name=name, args=args_list)))
  fn(obj)
}
```

Ecosystem interoperability in lcmsPlot

lcmsPlot's dependency stack

`lcmsPlot`	*your package (v0.99.20 at review)*
`patchwork`	*CRAN – figure composition*
`ggplot2`	*CRAN – plotting grammar*
`xcms`	*Bioc – chromatograms, spectra*
`Spectra, mzR`	*Bioc – MS data backends, raw file I/O*
`MsExperiment, MSnbase`	*Bioc – experiment container*
`BiocParallel, S4Vectors`	*Bioc core – parallelism, S4 foundation*

top row depends on rows beneath it

What the ecosystem costs you

- Your API must work with MsExperiment – not bespoke data frames
  When xcms releases, you test against the devel version
- Breaking changes in any upstream class ripple into your code
- Your release cadence is tied to Bioconductor's, not yours

What it buys you

- Nightly builds on three OSes you don’t have to maintain
- Landing page, BiocViews browser, support-site tag
- Discovery via the wider metabolomics ecosystem

General Bioconductor package development

- **R version**
  Always use the devel version of Bioconductor
- **Correctness**
  Minimally pass R CMD build and pass R CMD check with no errors and no warnings using a recent R-devel.
- **Correctness+**
  Pass BiocCheck::BiocCheckGitClone() and BiocCheck::BiocCheck('new-package'=TRUE) with no errors and no warnings.
- **Size**
  The source package resulting from running R CMD build should occupy less than 10 MB on disk. Individual file size less than 5 MB.
- **Duration**
  The package should require less than 10 minutes to run R CMD check --no-build-vignettes.

Code style and conventions

Prefer S4 over S3

Match the rest of the ecosystem. Define formal classes, generics, methods.

Use accessors, not @

Never @slot or slot() in exported code, examples or vignettes; expose getters/setters.

BiocParallel for parallelism

Don't use parallel::mclapply or the deprecated multicore. BiocParallel works on all OSes.

No browser() or print() leftovers

Use message() and warning(); reserve cat() for print-method output.

Documentation with roxygen2

```R
#’ Add a chromatogram layer to an lcmsPlot
#’
#’ Extracts and plots chromatograms over the
#’ supplied m/z and retention-time windows.
#’
#’ @param features A matrix or data frame with
#’   columns `mzmin`, `mzmax`, `rtmin`, `rtmax`.
#’ ...
#’
#’ @return An `lcmsPlot` layer object.
#’
#’ @examples
#’ data(example_lcms, package = "lcmsPlot")
#’ lcmsPlot(example_lcms) +
#’   chromatogram(features = rbind(c(
#’     mzmin = 300, mzmax = 305,
#’     rtmin = 2500, rtmax = 2550)))
#’
#’ @export
chromatogram <- function(features, ...) {
  # ...
}
```

What BiocCheck will check

- **@return on every page**
  Non-empty \value; otherwise WARNING.
- **@examples that run**
  At least 80% of man pages must have runnable examples.
- **@export for public API**
  Everything else stays internal; keep the surface small.
- **Consistent @param**
  Every argument documented; names must match.
- **Package-level page**
  ?lcmsPlot-package; a single overview Rd.

BiocCheck

R console — BiocCheck::BiocCheck('.')

```
> BiocCheck::BiocCheck("lcmsPlot")
— Checking Package DESCRIPTION ..... OK
— Checking biocViews ..... OK
— Checking version number ..... OK
! WARNING: Found > 80% of man pages need examples
           ('arrange' and 'facets' missing)
— Checking for direct slot access .....
✖ ERROR  : Found direct slot access via @ in
           vignettes/lcmsPlot.Rmd:142
— Checking unit tests ..... OK

Summary:  1 ERROR  |  1 WARNING  |  2 NOTE
See ./lcmsPlot.BiocCheck/ for the full report.
```

Severity levels

ERROR

Blocks acceptance. Fix these first.

WARNING

Must be fixed or justified before release.

NOTE

Informational. Often worth addressing anyway.

Install and run locally before submitting: `BiocManager::install("BiocCheck"); BiocCheck::BiocCheck(".", `new-package` = TRUE)`

Versioning

x . y . z

- **x — major** — Bumped only by Bioconductor team, at release, if y=99
- **y — branch** — Even = release, odd = devel. Set at release time
- **z — commits** — Bump this by 1 for every git commit

Submission workflow

1 **Pass all local checks**

R CMD build · R CMD check · BiocCheck – zero ERRORs, near-zero WARNINGs, justify any remaining NOTEs.

2 **Push to public GitHub (default branch)**

Source must be in the default branch of a public repo. Add SSH keys to your GitHub account; needed on git.bioconductor.org.

3 **Open an issue on Bioconductor/Contributions**

Title = package name, body = repo link + confirmation you've read the guidelines. You (the submitter) must be the maintainer.

4 **Pre-check + automated build**

Bot builds your package on Linux, macOS, Windows using R-devel. A build report is posted back on the issue.

5 **Peer review**

A Bioconductor reviewer reads your code, vignette, tests. Iterate by pushing commits (bump z each time) and replying on the issue.

6 **Acceptance → git.bioconductor.org**

Your package is copied to the Bioconductor git server and joins the nightly devel build. First full release at the next cycle.

Pre-submission checklist

```bash
# 1. Use the right R, match Bioconductor devel
R --version

# 2. Install Bioconductor devel
install.packages("BiocManager")
BiocManager::install(version = "devel")

# 3. Install checking tooling
BiocManager::install(c("BiocCheck", "biocthis"))

# 4. Run the two checks reviewers will run
BiocCheck::BiocCheckGitClone(".")
BiocCheck::BiocCheck(".", `new-package` = TRUE)

# 5. Confirm build size is under 5 MB
R CMD build lcmsPlot
ls -lh lcmsPlot_0.99.0.tar.gz

# 6. Add an SSH public key to your GitHub account
#    (will be reused for git.bioconductor.org later)
```

Whitelist Bioconductor senders

Let your mail client accept noreply@bioconductor.org and BBS-noreply@bioconductor.org. Build-failure emails land there.

Register on the Support site

support.bioconductor.org – create an account and add your package name to your watched tags.

README.md has a BiocManager block

Include BiocManager::install("lcmsPlot") as the canonical install. If README.Rmd, set eval=FALSE.

Confirm SSH key

Same key works for GitHub and later for git.bioconductor.org. Test with ssh -T git@github.com.

What the review looks like

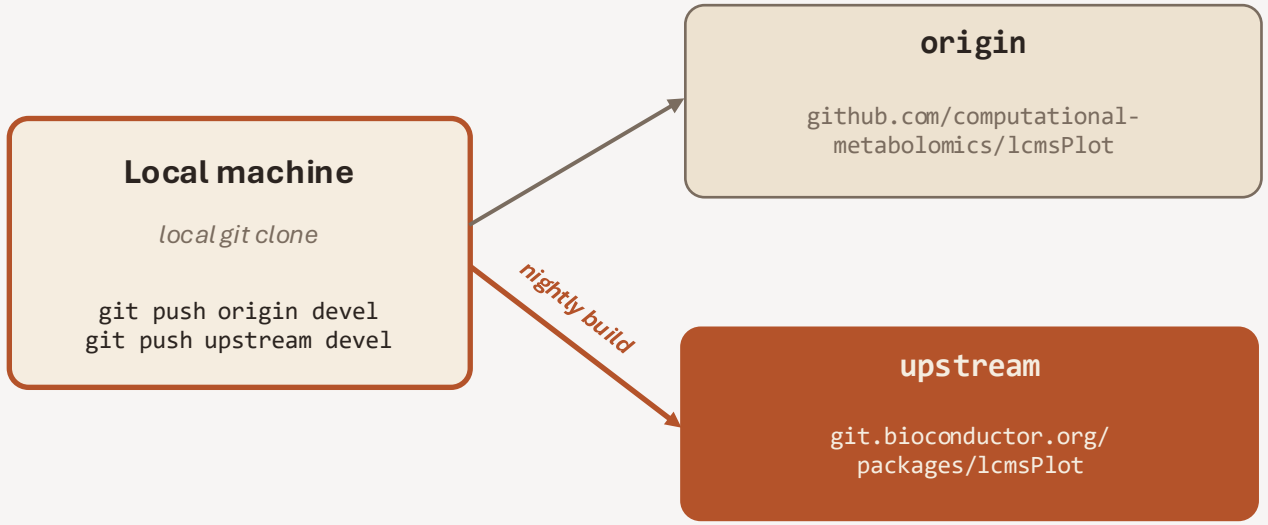
Label lifecycle on your submission issue

1. awaiting moderation → 1a. pre-check passed → 2. review in progress → 3. review done → 3a. accepted

What reviewers look at

Required	Recommended	Suggested
Must be addressed before acceptance.	*Address or justify in a reply.*	*Take it or leave it, your call.*
• Any remaining BiocCheck ERROR	• BiocCheck WARNINGs you skipped	• Better function names
• Direct @ slot access in examples	• Man-page examples < 80% coverage	• Additional vignette sections
• Missing biocViews or wrong category	• Non-exported helpers documented	• Coverage or benchmark improvements
• Vignette that doesn't build	• Coding style inconsistencies	• Reduce the number of exported functions

After acceptance




```bash
# Add the Bioconductor git server as a second remote
git remote add upstream git@git.bioconductor.org:packages/lcmsPlot.git
git fetch upstream
git merge upstream/devel --allow-unrelated-histories
# From now on, push every change to BOTH remotes
git push origin devel && git push upstream devel
```

1. **Nightly build ingests your code**
   Every push to upstream triggers a rebuild across Linux, macOS, Windows.

2. **Landing page goes live**
   bioconductor.org/packages/devel/bioc/html/lcmsPlot.html with DOI, vignette, install instructions.

3. **Status badges turn green**
   Users see your build status on the landing page.

4. **You inherit a Support-site tag**
   Questions tagged lcmsPlot show up on support.bioconductor.org.

You're a maintainer now!

Daily: check build status

Bookmark bioconductor.org/checkResults/devel/bioc-LATEST/lcmsPlot: fix red badges same-week

Per commit: bump z + update NEWS.md

Every push to upstream is a new version. No bump = the Bioconductor repo ignores your change.

Weekly: scan the support site

Reply to tagged questions within a few days. Silence is a red flag to the core team.

Monthly: skim bioc-devel

Low-traffic list. Major ecosystem changes are announced here: breaking changes in xcms, MSnbase, etc.

Where to get help

`contributions.bioconductor.org`

The full developer handbook

`bioc-devel mailing list`

Technical questions during submission

`support.bioconductor.org`

User-facing Q&A; watch your package tag

`github.com/Bioconductor/Contributions`

Where submissions live

`BiocCheck vignette`

Decode every ERROR/WARNING/NOTE