

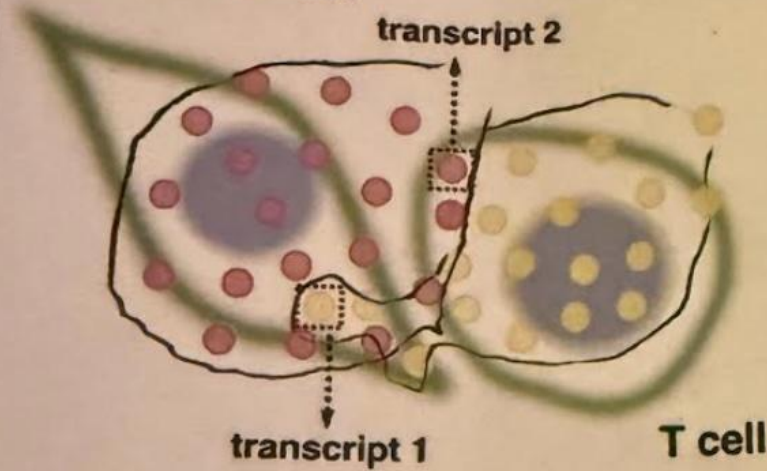
# Bioconductor and LLMs, CSAMA 2026

VJ Carey

# Strategic thoughts

- "Statistics is the most successful information science. Those who ignore it are condemned to reinvent it." -- possibly B. Efron
- Variability is the focus of statistical reasoning
  - make friends with a statistician, biostatistician
  - involve them early on in your design considerations
  - ALWAYS REPORT DETAILS ON
    - "SAMPLE SIZE"
    - INDEPENDENCE OR DEPENDENCE (e.g., hierarchical structure)
    - UNCERTAINTY (e.g., confidence interval, FDR)
  - When these are absent we have a problem
- As "models" get more complex, for a fixed sample size, a price must be paid -- "degrees of freedom" are lost

Fibroblast



- CD3
- COL1A1
- Boundary stain
- Nuclear stain

	T cell	Fibroblast
CD3		
COL1A1		

## Questions

- 1) How would you delineate each cell entity?
- 2) Which cells do transcript 1 and 2 belong to?
- 3) What is the morphology of the two cells (round, elongated)?
- 4) What does the count matrix look like?

T (1)      fib (2)  
T is elong  
fib round

Please produce R code that creates a quasi-rhomboid shape in the unit square by joining the traces of a parabolic function and its reflection about the 45 degree line. The structure has "vertices" at northeast and southwest corners of the unit square and the sharpness of curves away from the vertices is determined by a scalar parameter.

Here's the R code. The shape is formed by a parabola  $y = x^k$  on  $[0,1]$  and its reflection  $x = y^k$  (equivalently  $y = x^{(1/k)}$ ), meeting at  $(0,0)$  and  $(1,1)$ . The parameter  $k > 1$  controls curvature — larger  $k$  gives sharper indentations at the midpoints, making the shape more "pinched."

Created a file, read a file >

Now let me also show a preview of what the shape looks like:



Great. Now can this concept be used to produce a collection of 50 traces of randomly "sized" and randomly rotated quasi-rhomboids with different values of  $k$ , in a planar square of side length 5 units?

Ran 3 commands, viewed a file, read a file >

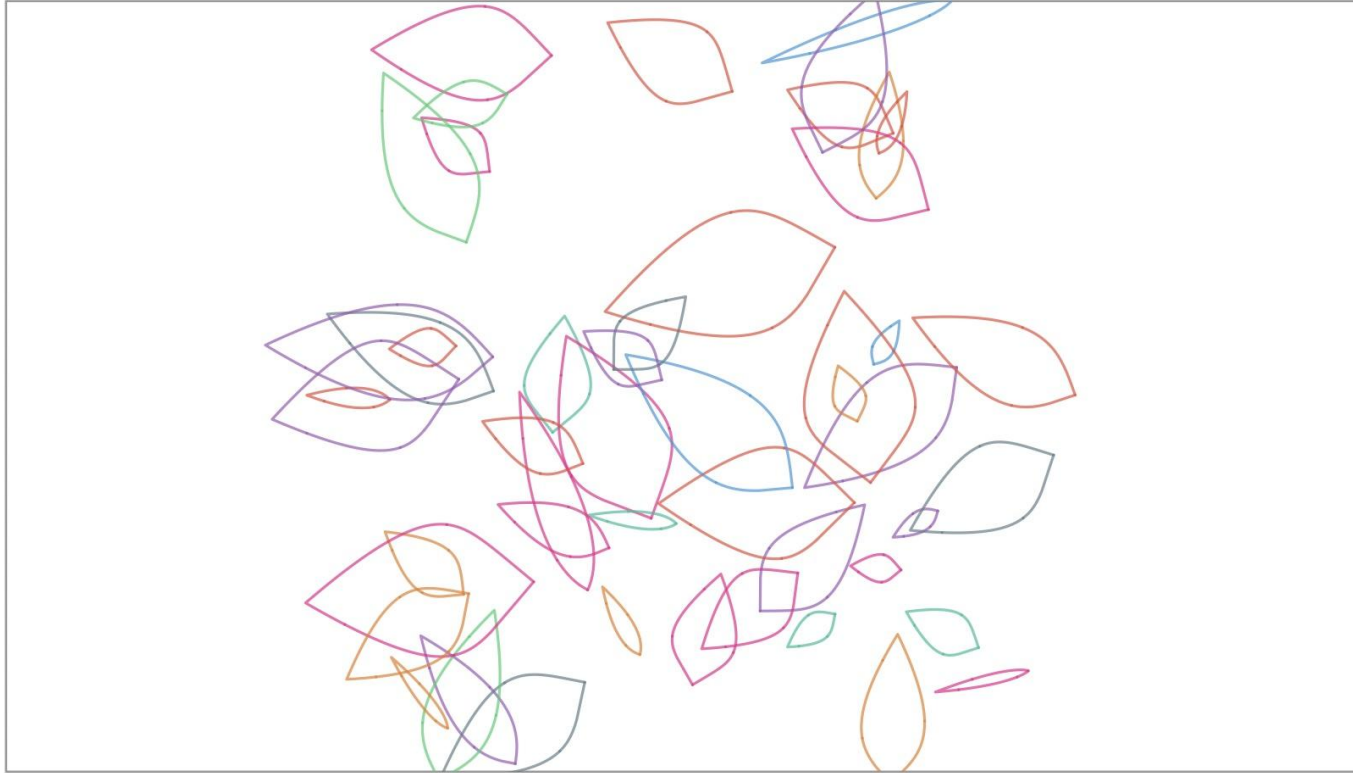
Here is the new script. The key ideas:

### Coordinate pipeline for each shape:

1. `rhomboid_poly(k)` samples the closed boundary in the unit square.
2. The polygon is **centred** by shifting by `(-0.5, -0.5)` so the geometric centre sits at the origin.
3. It is **scaled** by a random `size` drawn from `Uniform(0.25, 1.2)` world units.
4. It is **rotated** by multiplying through a  $2 \times 2$  rotation matrix with a random angle from `Uniform(0, pi)` — a half-turn suffices because the shape has 180-degree symmetry.
5. It is **translated** to a random centre `(cx, cy)` drawn uniformly from `[0, 5]^2`.

Shapes are drawn as outlines only (`lines()` with 70% alpha) so overlapping regions stay legible rather than turning into an opaque pile. Adjust `sizes`, `alpha`, or `lwd` to taste, or add a `polygon()` fill call if you want colour masses instead.

## 50 random quasi-rhomboids



What are next steps in making this procedure more realistic for simulating cell boundaries in a "slice"?

# Road map

- remote (e.g., anthropic) and local (e.g. ollama) models
- using LLMs within R - why?
  - "converse" without jumping out of your analysis environment
  - "perform computations" by prompting the LLM
  - "generate code" by prompting the LLM - simulating spatial patterns by asking
- prompting for structured response - classifying submitted packages
- agentic coding first steps
  - "skills" documents that establish context
  - evaluation of model performance for code generation when guided by skill document



Hi, suppose I have produced a nice DT::datatable object in R ... how can I serialize it for inclusion in a web page, preferably with a given sort order?

Deciphered serialization methods for DT::datatable web integration >

Great use case. Here are the main approaches:

### 1. Save as a standalone HTML widget (most common)

```
r  
  
library(DT)  
library(htmlwidgets)  
  
dt <- datatable(mtcars, options = list(order = list(list(1, 'asc'))))  
  
saveWidget(dt, "my_table.html", selfcontained = TRUE)
```

`selfcontained = TRUE` bundles all JS/CSS inline — one portable file. Use `FALSE` if you want a lighter HTML file with a `_files/` directory alongside it.



### 2. Extract just the widget snippet (for embedding in an existing page)

Write a message...



Sonnet 4.6 Adaptive ▾



# Claude code CLI -- assess cost of iterative creation of a shiny app

```
- Claude.ai subscription (Pro/Max) – Claude Code usage is billed through your subscription, not the API. It won't appear in console.anthropic.com API usage dashboards; you'd see it in your claude.ai account billing instead.  
- API key – If Claude Code is configured with an Anthropic API key (ANTHROPIC_API_KEY), usage should appear in the console under your API project.
```

```
You can check which mode you're in by running ! claude config or looking at your Claude Code settings – if there's no API key configured, you're running on the subscription plan and that's why the console shows nothing.
```

```
Now, back to the mt-Co3 problem – should I run a Bash command to inspect the plotShape source, or would you prefer to share how plotShape handles the fill argument?
```

```
* Crunched for 12s
```

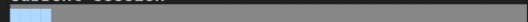
```
/cost
```

```
Settings Status Config Usage Stats
```

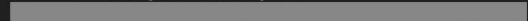
## Session

```
Total cost:           $0.59  
Total duration (API):  4m 36s  
Total duration (wall): 3h 16m 32s  
Total code changes:   106 lines added, 4 lines removed  
Usage by model:  
  claude-haiku-4-5:    453 input, 20 output, 0 cache read, 0 cache write ($0.0006)  
  claude-sonnet-4-6:   1.5k input, 15.1k output, 610.5k cache read, 45.8k cache write ($0.59)
```

## Current session

```
 8% used  
Resets 4pm (Europe/Rome)
```

## Current week (all models)

```
 0% used  
Resets Jun 4 at 12pm (Europe/Rome)
```

## What's contributing to your limits usage?

```
Approximate, based on local sessions on this machine – does not include other devices or claude.ai
```

```
Last 24h · these are independent characteristics of your usage, not a breakdown
```

```
Nothing over 10% in this period – try the other window.
```

```
d to day · w to week
```

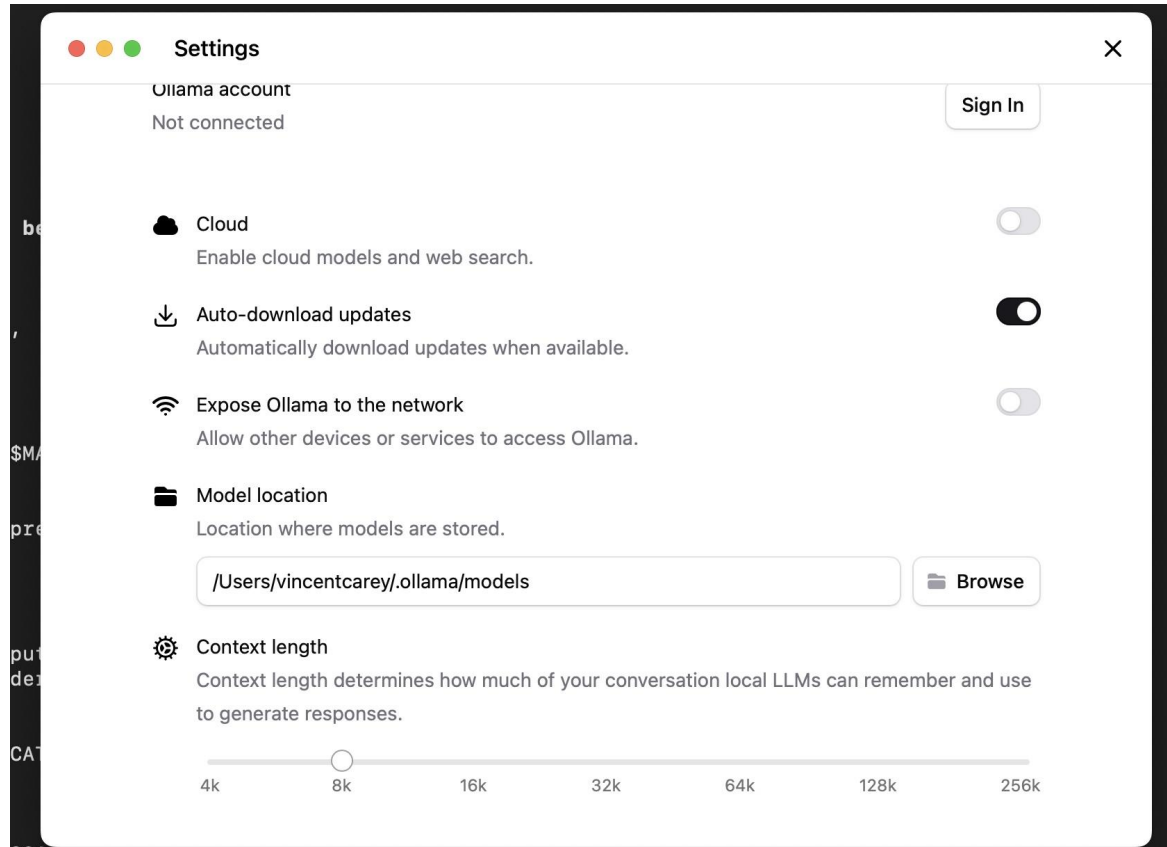
```
Esc to cancel
```

Screensh

# Upshots

- "Chatbot" simplifies discovery and can write code that you can inspect, copy, download
- "Agent" interacts directly with file system, can edit files, make commits and pull requests
- A tutorial on agentic coding from Sean Davis, friend of Bioconductor, is [available](#)
- Tutorial includes discussion of costs; introspection on resource consumption seems reasonable

# Local LLM on laptop via ollama



# Models I happen to have -- 30b qwen-coder just fits?

```
PC002566:~ vincentcarey$ ollama list
```

NAME	ID	SIZE	MODIFIED
qwen3-coder:30b	06c1097efce0	18 GB	2 months ago
nomic-embed-text:latest	0a109f422b47	274 MB	2 months ago
gpt-oss:20b	17052f91a42e	13 GB	5 months ago
unclemusclez/jina-embeddings-v2-base-code:latest	9fe680d4d58b	322 MB	6 months ago
jina/jina-embeddings-v2-base-en:latest	377193292084	274 MB	6 months ago
qwen3:8b	500a1f067a9f	5.2 GB	6 months ago
qwen3:4b	359d7dd4bcda	2.5 GB	6 months ago
gemma3n:latest	15cb39fd9394	7.5 GB	10 months ago
qwen2.5vl:latest	5ced39dfa4ba	6.0 GB	10 months ago
deepseek-r1:14b	c333b7232bdb	9.0 GB	11 months ago
llama3.2:3b-instruct-q4_K_M	a80c4f17acd5	2.0 GB	11 months ago

# Channeling Daria's quiz a bit

r = "Write R code to produce coordinates in the plane of 15 points that are uniform in the unit circle. Then produce a rotation matrix that transforms the positions of these points to an ellipse with major axis at 135 degrees and minor axis at 45 degrees, with minor axis .5 the length of major axis."

```
library(ellmer)
```

```
ch = chat_ollama(model = "qwen3-coder:30b")
```

```
ch$chat(pr)
```

```
# resulting code is in a gist
```

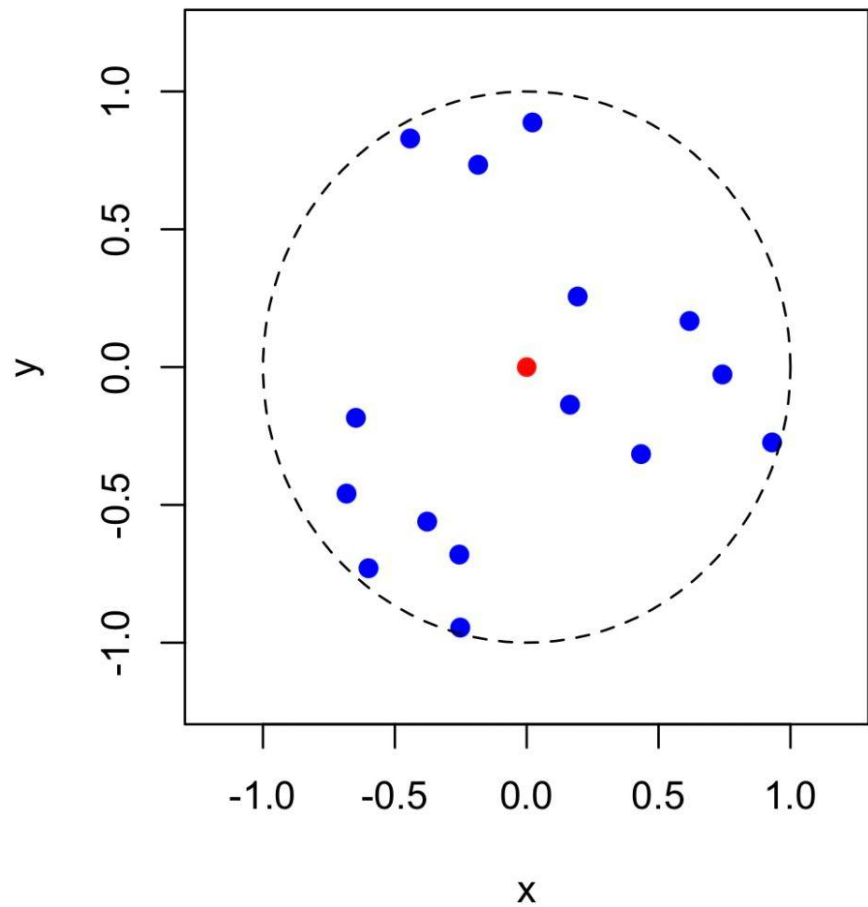
	T cell	Fibroblast
CD3		
COL1A1		

**Questions**

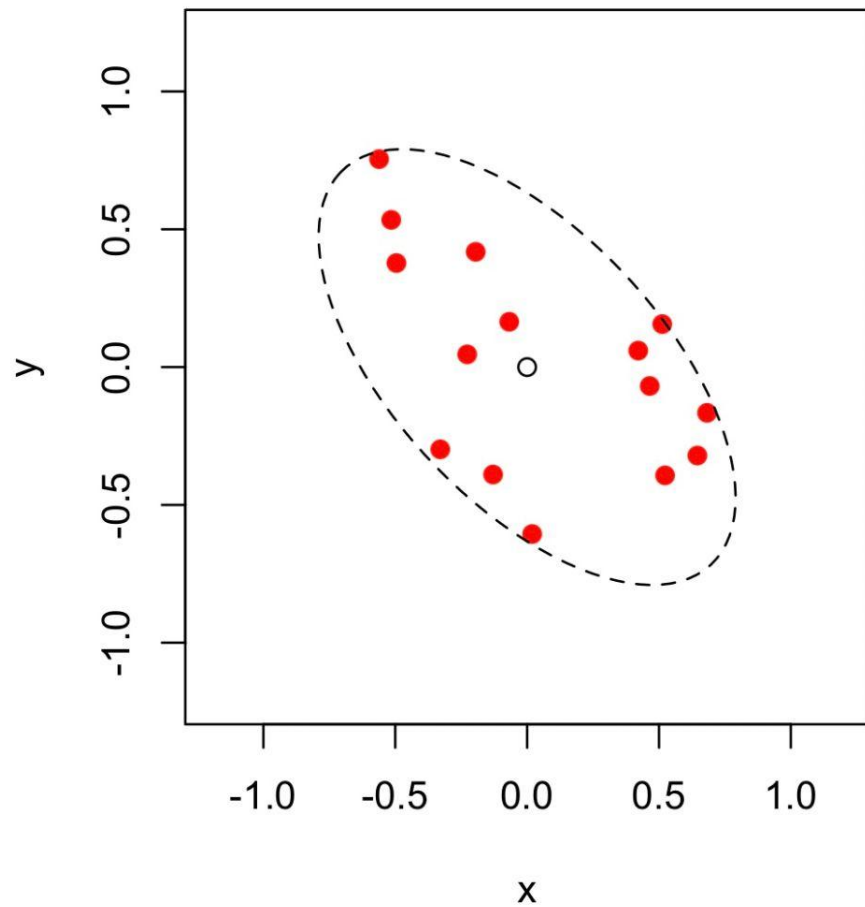
- 1) How would you delineate each cell entity?
- 2) Which cells do transcript 1 and 2 belong to?
- 3) What is the morphology of the two cells (round, elongated)?
- 4) What does the count matrix look like?

*T(1) Fib(2)  
T is elong  
Fb round*

### Original Points in Unit Circle



### Transformed Points as Ellipse



# Upshots

- Local, on-laptop code generation
- Went beyond the request, produced the data and unrequested visualization support
- Exercise: use another model to critique/improve the code
- Exercise: specify detailed simulation of transcript/stain patterns in 3-d, use as "ground truth" for segmentation and cell type inference evaluation for slices
- I didn't do it yet, but consider the following as a simulation of a certain family of cell shapes in the plane

# Structured query output, from vig2data

```
type_summary <- type_object(.description = "Summary of the article.",
  author = type_array(description = "Name of the article author(s)",
    type_string(), ), topics = type_array(description = "Array of topics in the biosciences, e.g. [\"DNA\",
  \"RNA\", \"chromosomal positions\", \"genes\"]. Should be as specific as possible, and can overlap.",
    type_string()), focused = type_string(description = "Provide a concise summary of the article using
  distinctive vocabulary that would be helpful for embedding the summarized content. Avoid subjective
  judgmental commentary. Avoid generalities about hypotheses and generic activities of data analysis.
  Limit the summary to 100 words. Markdown annotations should be as simple as possible. Do not use
  asterisks for emphasis."),
  coherence = type_integer("Coherence of the article's key points, 0-100 (inclusive)"),
  persuasion = type_number("Article's persuasion score, 0.0-1.0 (inclusive)"))
chat <- chatfunc(model = model, ...)
chat$chat_structured(substr(text, 1, maxnchar), type = type_summary)
```

Full table  
[available](#)

github.com/vjcitnr/biocSubmitted/blob/main/packages\_classified.md

Emily Osterweil | O... OSF | Bulker: a mu... Agentic systems a... Rashid Lab Website GitHub - seandavi/... VeniceHackathon2... Problem set 7 - B... >>

main ▾ biocSubmitted / packages\_classified.md

Code Blame 191 lines (154 loc) · 95.4 KB

## Single-Cell and Spatial Transcriptomics

*Packages for single-cell RNA-seq analysis, spatial transcriptomics, cell type annotation, and related experimental data containers like SingleCellExperiment and SpatialExperiment*

Package Authors	Description	Key Topics
Harriet Jiali He, Stephanie C. Hicks	SpatialArtifacts is a Bioconductor R package implementing a two-step workflow (detection and classification) to identify and categorize spatial artifacts in spatial transcriptomics data from 10x Visium and VisiumHD platforms. The method combines median absolute deviation-based outlier detection with morphological image processing using focal transformations to detect edge artifacts (low gene/UMI counts at tissue boundaries) and interior artifacts. The detectEdgeArtifacts function identifies problem areas through outlier clustering and boundary evaluation, while classifyEdgeArtifacts applies hierarchical logic based on location and size to assign labels (large/small edge/interior artifacts). Platform-specific parameters accommodate different grid arrangements (hexagonal for Visium, square for VisiumHD) and resolutions (55µm, 16µm, 8µm bins), with automatic scaling for physical unit consistency across platforms.	spatial transcriptomics, 10x Visium, VisiumHD, quality control, artifact detection, morphological image processing, SpatialExperiment, Bioconductor, UMI counts, mitochondrial ratio, tissue edge artifacts, interior artifacts, median absolute deviation, outlier detection
Nick Borcherding	immLynx is an R package integrating Python-based immune repertoire analysis tools including tcrdist3 for TCR distance calculations, OLGA for generation probability, soNNia for selection inference, clusTCR for clustering, and ESM-2 protein language models for embeddings. The package operates on SingleCellExperiment objects containing TCR data from scRepertoire. Functions extract and validate TCR sequences, cluster receptors using MCL algorithm, calculate pairwise distances, compute Pgen values, generate embeddings for dimensionality reduction, and identify metaclones. The package uses basilisk for automated Python dependency management and provides unified interfaces compatible with single-cell immune repertoire workflows.	T-cell receptors, TCR repertoire, immune repertoire sequencing, protein language models, single-cell sequencing, TCR clustering, sequence generation probability, TCR distance metrics, metaclones, protein embeddings, CDR3 sequences

# Upshots

- Submissions arrive rapidly; authors may not concisely describe their packages
- Purely administrative tasks: extract/infer keywords, manage contributor email, present summaries in HTML, match to available reviewers - \$10/run on 100 submissions?
- ellmer's structured query management plus R programming help to characterize packages in queue
  - prioritization?
  - reduction of redundancy among submissions? pairing devs working on similar projects?












# Agentic coding - not just prompting for code

"Skills" that enhance prompt interpretation have been developed.

"how-to"  
for analyst

Name	Last commit message
..	
compute-read-coverage.md	first draft with worked evaluation computations
compute-sequence-composition-for-genomic-regions.md	first draft with worked evaluation computations
extract-promoter-sequences.md	first draft with worked evaluation computations
get-exon-intron-sequence-for-gene.md	first draft with worked evaluation computations
load-gene-from-gff-gtf.md	first draft with worked evaluation computations
read-big-bam-file-in-chunks.md	first draft with worked evaluation computations
read-gene-sets-from-gmt-files.md	first draft with worked evaluation computations
read-mass-spectrometry-data.md	first draft with worked evaluation computations
read-paired-end-reads-from-bam-file.md	first draft with worked evaluation computations
read-single-end-reads-from-bam-file.md	first draft with worked evaluation computations
retrieve-gene-model-from-annotationhub.md	first draft with worked evaluation computations
use-tidy-principles-for-granges-manipulation.md	first draft with worked evaluation computations
use-tidy-principles-for-rna-seq-analysis.md	first draft with worked evaluation computations

# Skills for developers, skill-writers, ...

Name	Last commit message
 ..	
 analyze-r-package	first draft with worked evaluation computations
 check-waldronlab-skills	first draft with worked evaluation computations
 create-package-instructions	first draft with worked evaluation computations
 create-skill	first draft with worked evaluation computations
 document-skill	first draft with worked evaluation computations
 improve-code-coverage	first draft with worked evaluation computations
 security-audit-r-package	first draft with worked evaluation computations
 update-package-instructions	first draft with worked evaluation computations
 update-r-news	first draft with worked evaluation computations
 validate-skill	first draft with worked evaluation computations

<b>name</b>	load-gene-from-gff-gtf
<b>description</b>	Import a gene model from a GFF or GTF file as a TxDb object using Bioconductor

## Load a Gene Model from a GFF or GTF File

A **gene model** describes the genomic locations of genes, transcripts, exons, and CDS. Use `makeTxDbFromGFF()` from the **txdbmaker** package to import a GFF3 or GTF file as a `TxDb` object, which is the standard Bioconductor representation for gene models.

### Required Packages

```
BiocManager::install("txdbmaker")  
  
library(txdbmaker)
```

### Steps

#### 1. Import a GFF/GTF file as a TxDb

```
# Replace with a path to your own GFF3 or GTF file  
gff_file <- "/path/to/annotation.gff3" # or .gtf  
  
txdb <- makeTxDbFromGFF(gff_file, format = "gff3")  
# Use format = "gtf" for GTF files; "auto" also works for auto-detection  
txdb
```

Using the example file bundled with txdbmaker:

```
gff_file <- system.file("extdata", "GFF3_files", "a.gff3", package = "txdbmaker")  
txdb <- makeTxDbFromGFF(gff_file, format = "gff3")
```

#### 2. Extract gene model features

```
# Exons grouped by gene  
exonsBy(txdb, by = "gene")
```

A skill file is markdown with instructions and example to guide the LLM. Skills can be located in specific positions on your computer to drive interactions with different LLMs.

To assess the effectiveness of a skill for a class of tasks to be performed with a given model,

- "ground truth" facts are collected,
- task-prompted code is executed, and
- results are compared to the facts.

```
=== Static checks ===
      check status detail
      uses txdbmaker PASS
avoids deprecated path PASS
      format= arg PASS
      exonsBy PASS
      transcriptsBy PASS
      cdsBy PASS

=== Correctness checks ===
      check status detail
      is TxDb PASS
      gene count PASS 488 (expected 488)
transcript count PASS 488 (expected 488)
      exon count PASS 1268 (expected 1268)
      CDS groups PASS 488 (expected 488)
      seqlevels PASS 'SL2.40ch00' (expected 'SL2.40ch00')
      gene1 start PASS 16437
      gene1 end PASS 18189
      gene1 strand PASS +
      gene1 exon count PASS 2
      gene1 exon1 start PASS 16437
      gene1 exon1 end PASS 17275
      gene1 exon2 start PASS 17336
      gene1 exon2 end PASS 18189
      gene1 tx count PASS 1
      gene1 tx name PASS Solyc00g005000.2.1
      gene3ex exon count PASS 3 (expected 3)
      gene3ex exon starts PASS 68062,68344,68654
      gene4ex exon count PASS 4 (expected 4)
      gene5ex exon count PASS 5 (expected 5)
      gene5ex strand PASS -

Overall: PASS (27/27 checks passed)
```

## Illustration of “prompt ladder”

There are levels of sophistication in prompting the LLM that Claude has proposed for evaluating skills. In this example there are 5 levels of prompts related to extracting gene models from GFF or GTF:

Prompt	Framing	Expected failure mode
<b>P1</b> Minimal	“Load a GFF file in R and show me the genes and exons.”	Uses <code>rtracklayer</code> instead of <code>txdbmaker</code> ; no TxDb
<b>P2</b> Package hint	“Use txdbmaker to load a GFF3 file...”	Correct package, may miss <code>format=</code> or <code>system.file()</code>
<b>P3</b> Skill-aligned	Names the function, the bundled file path, and asks for all three grouping calls	Near-reference; should pass most checks
<b>P4</b> Full skill doc	Skill how-to injected as system prompt	Highest fidelity; should match reference exactly
<b>P5</b> Adversarial	Asks for <code>GenomicFeatures::makeTxDbFromGFF</code> explicitly	Tests whether skill overrides legacy framing

# Conclusions

- IMHO, "data science" and "genome biology" both suffer from LOW VERIFICATION BANDWIDTH
- Data volume and complexity were the origins of Bioconductor: biologists and statisticians had to unite to tackle DNA MICROARRAYS!
- Consistency and stability of Bioconductor remain important values
- Use parallel computing and trained agents to IMPROVE VERIFIABILITY OF OUR METHODS AND RESULTS