

UC San Diego

UC San Diego Previously Published Works

Title

Preparing glycomics data for robust statistical analysis with GlyCompareCT

Permalink

<https://escholarship.org/uc/item/87h3v5wz>

Journal

STAR Protocols, 4(2)

ISSN

2666-1667

Authors

Zhang, Yujie
Krishnan, Sridevi
Bao, Bokan
et al.

Publication Date

2023-06-01

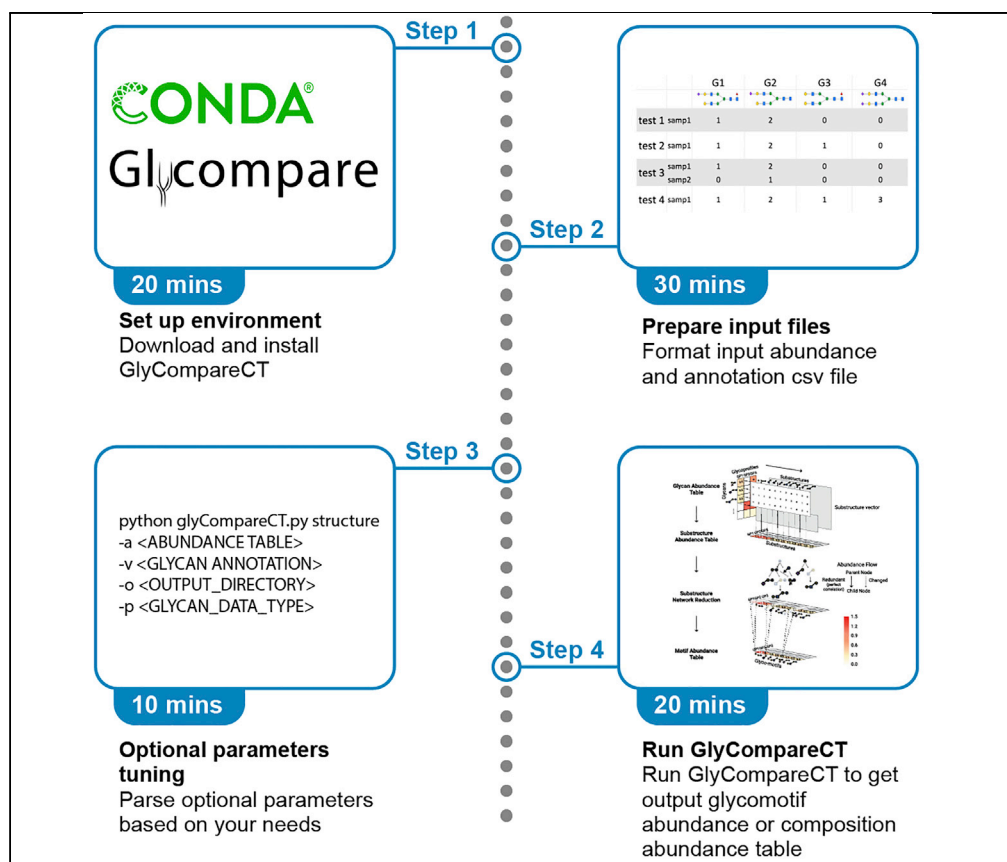
DOI

10.1016/j.xpro.2023.102162

Peer reviewed

Protocol

Preparing glycomics data for robust statistical analysis with GlyCompareCT



Yujie Zhang, Sridevi Krishnan, Bokan Bao, ..., Song-Min Schinn, Benjamin P. Kellman, Nathan E. Lewis

benjamin.kellman@gmail.com (B.P.K.)
nlewisres@ucsd.edu (N.E.L.)

Highlights

Bioinformatics tool for processing glycomic data sets

Glycan structure decomposition to increase statistical power and increase interpretability

Easy-to-use command line executables

GlyCompareCT is a portable command-line tool to facilitate downstream glycomic data analyses, by addressing data inherent sparsity and non-independence. Inputting glycan abundances, users can run GlyCompareCT with one line of code to obtain the abundances of a minimal substructure set, named glycomotif, thereby quantifying hidden biosynthetic relationships between measured glycans. Optional parameters tuning and annotation are supported for personal preference.

Publisher's note: Undertaking any experimental protocol requires adherence to local institutional guidelines for laboratory safety and ethics.

Zhang et al., STAR Protocols 4, 102162
June 16, 2023 © 2023 The Author(s).
<https://doi.org/10.1016/j.xpro.2023.102162>



Protocol

Preparing glycomics data for robust statistical analysis with GlyCompareCT

Yujie Zhang,^{1,2,3} Sridevi Krishnan,¹ Bokan Bao,^{1,3} Austin W.T. Chiang,¹ James T. Sorrentino,^{1,3} Song-Min Schinn,^{1,3} Benjamin P. Kellman,^{1,3,4,5,6,*} and Nathan E. Lewis^{1,3,5,7,*}

¹Department of Pediatrics, University of California, San Diego, 9500 Gilman Drive MC 0760, La Jolla, CA 92093, USA

²Department of Biostatistics, Harvard T.H. Chan School of Public Health, 677 Huntington Avenue, Boston, MA 02115, USA

³Department of Bioengineering, University of California, San Diego, 9500 Gilman Drive MC 0760, La Jolla, CA 92093, USA

⁴Augment Biologics, 9450 SW Gemini Dr. #46664, Beaverton, OR 97008, USA

⁵These authors contributed equally

⁶Technical contact: benjamin.kellman@gmail.com

⁷Lead contact

*Correspondence: benjamin.kellman@gmail.com (B.P.K.), nlewisres@ucsd.edu (N.E.L.)
<https://doi.org/10.1016/j.xpro.2023.102162>

SUMMARY

GlyCompareCT is a portable command-line tool to facilitate downstream glycomic data analyses, by addressing data inherent sparsity and non-independence. Inputting glycan abundances, users can run GlyCompareCT with one line of code to obtain the abundances of a minimal substructure set, named glycomotif, thereby quantifying hidden biosynthetic relationships between measured glycans. Optional parameters tuning and annotation are supported for personal preference. For complete details on the use and execution of this protocol, please refer to Bao et al. (2021).¹

BEFORE YOU BEGIN

Overview

Glycosylation is a hierarchical process complicated by numerous processive, and high-specificity biosynthetic enzymes that compete for common substrates.² Innovations in mass spectrometry enabled rapid growth in the number of discovered glycans.^{3–12} Comprehensive, and consistent analytics are increasingly critical as rich glycoprofile datasets are cataloged into ever-larger datasets and databases.^{13–19} Sparsity of glycan profiles across observations and structural commonalities between different glycans (non-independence) create challenges in rapid and accurate comparison of glycoprofiles. GlyCompareCT solves these challenges by decomposing glycan structures into substructures and identifying all glycomotifs, which are the minimal set of the largest non-redundant substructures sufficient to describe all variance in the dataset. Procedures for the installation and formatting of the input files are presented here, followed by detailed guidance in the parameterization and use of GlyCompareCT.

Performance and comparison with GlyCompare

GlyCompare is the Python library that GlyCompareCT builds on. GlyCompareCT wraps the GlyCompare intermediate steps to an integrated pipeline and improves implementation efficiency. To provide an example of expected tool performance, We present a performance comparison between GlyCompareCT's and GlyCompare v1.0.0 across several glycan datatypes.

We tested GlyCompareCT on erythropoietin N-glycosylation,^{20,21} human milk oligosaccharides,²² mucin-type O-glycans,^{23,24} and gangliosides²⁵ with default parameters. We first tested the change



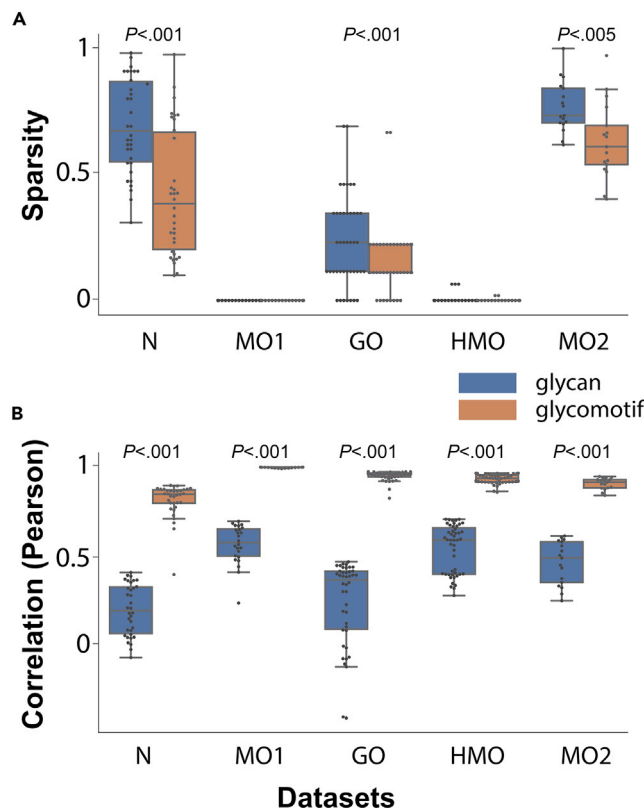


Figure 1. GlyCompareCT sparsity and correlation change

GlyCompareCT was tested on erythropoietin N-glycosylation (N), human milk oligosaccharides (HMO), mucin-type O-glycans (MO), and gangliosides (GO) datasets.

(A) Boxplots show that glycoprofile sparsity (zero-abundance fraction) decreased in three of five datasets after substructure decomposition to glycomotif-profiles (KS tests, $p < 0.001$ for N and GO, and $p < 0.005$ for MO2).

(B) Boxplots show inter-profile correlation increases between glycomotif profiles after substructure decomposition (KS tests, $p < 0.001$ for all datasets). Boxplots (a-b) display the median (centerline), interquartile range (IQR; box), and $1.5 \times \text{IQR}$ (whiskers).

in sparsity in each dataset. In three of five datasets, glycomotif abundance tables were less sparse than corresponding glycan abundance tables (Figure 1A). In the remaining two datasets, glycan abundance sparsity is already close to zero (an artifact of targeted measurement of select glycans); expected when input glycan abundance table sparsity is already close to zero (Figure 1A). Furthermore, we observe that correlation between glycomotif profiles is higher than correlation between glycan abundance profiles across all datasets (Figure 1B).

To demonstrate parameter generalizability, we tested all combinations of all parameters on the datasets. Regardless of parameterization, sparsity decreases (KS-tests, $p < 0.02$) and correlation increases (KS-tests, $p < 0.001$).

These examples show that users of GlyCompareCT are expected to see decreased sparsity and increased correlation, consistent with that seen with GlyCompare.¹ Furthermore, by decreasing sparsity and increasing correlation, GlyCompareCT can make glycoprofile interdependence explicit. It thereby increases statistical power, helps to elucidate trends, and support statistical comparisons.

We compared GlyCompareCT v1.1.3 with GlyCompare v1.0.0,¹ the Python library it builds on. To show the runtime and memory change between GlyCompare and GlyCompareCT, we ran both tools

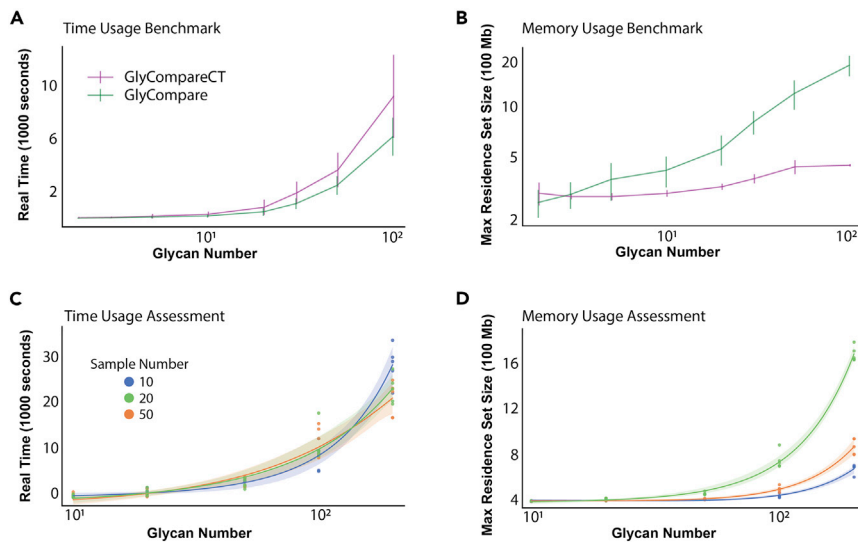


Figure 2. GlyCompareCT Performance and Comparison with GlyCompare

(A and B) Performance improvements in GlyCompareCT v1.0.0 over Glycompare v1.1.3. (A) Line plots with error bars (standard deviation of 10 independent datasets) indicate minimal changes to runtime. (B) Line plots with error bars (standard deviation of 10 independent datasets) indicate substantial improvements in memory. (C and D) Time and memory usage performance on varying numbers of glycans and samples. Each pair of (glycan_num, sample_num) contains five trials. (C) Line plots are linear regression lines fitted to runtime. (D) Line plots are linear regression lines fitted to memory.

using 32 CPUs on 80 unit-datasets (wherein all glycan abundances are 1, each dataset contains 1 sample, and 2, 3, 5, 10, 20, 30, 50, or 100 glycans chosen randomly from the linkage-specific glycomotif reference [github.com/yuz682/GlyCompareCT/tree/main/reference]). We note that high-resource computers (e.g., 32 CPUs) will not be necessary for most datasets. We examined 10 datasets for each data size. GlyCompareCT shows a modest but significant time increase compared to GlyCompare (1.5–1.7 fold; 2-sample 2-sided t-test, $p < 0.02$, Figure 2A). GlyCompareCT also shows a substantial maximum resident set size (RSS) decrease from GlyCompare. Analyzing unit-datasets with 100 randomly selected glycans, GlyCompareCT approaches a 5-fold reduction in memory (from 1,923 Mb to 446 Mb, Figure 2B). To further assess the runtime and memory performance of GlyCompareCT, we tested multiple datasets varying glycan number and sample number, including 10, 20, 50, 100, and 200 unique glycans represented in 10, 20, and 50 samples. For each pair of glycan number and sample number, we ran five trials. For each trial, we chose glycans randomly and randomly set 80% of entries to 0 and the remaining 20% of entries to 1—simulate the sparsity in real glycoprofiles. The time usage does not vary significantly across different sample numbers (Figure 2C) while the memory usage increases with more samples (Figure 2D).

The steps for using GlyCompareCT now follow.

Downloading GlyCompareCT

⌚ Timing: < 5 min

To get started quickly, GlyCompareCT can be downloaded as a precompiled executable (see Optional below). To run GlyCompareCT directly in Python (operating system agnostic), users can download executable Python scripts.

Optional: GlyCompareCT command line executables are precompiled for the Mac, Linux and Windows operating systems and available on Zenodo <https://zenodo.org/record/6370789#>.

Y6eNrOzMIUE and GitHub <https://github.com/LewisLabUCSD/GlyCompareCT/releases/tag/v1.2.0>. The executables are ready-to-use once downloaded. Executables will be maintained for large updates, while Python scripts on GitHub might undergo small fixes more often.

1. **Command line python scripts** for GlyCompareCT at Github repo LewisLabUCSD/GlyCompareCT. Using Python scripts requires Python environment setup. Download these scripts either by cloning the Github repo or from the Github release.
 - a. To download the stable release (v1.2.0) of command line python scripts, install from Github release. Users can go to <https://github.com/LewisLabUCSD/GlyCompareCT/releases/tag/v1.2.0> and click to download the Python script contained in the Source Code zipped file.
 - b. To download the development version of command line python scripts, users can download the GlyCompareCT Github repo, which requires git on your local machine (check this [instruction](#) if you need help install git). On Linux, MacOS (terminal) or Windows (PowerShell or Anaconda).

```
>git clone https://github.com/LewisLabUCSD/GlyCompareCT.git
```

Environment setup

⌚ Timing: < 20 min

Executables are ready to use once downloaded while Python scripts require Python environment setup.

Note: All codes are run in Linux, MacOS terminal, or Windows PowerShell. ">" indicates a new line and should not be typed.

Optional: The users can directly use the executables once downloaded. See [troubleshooting 1](#) if needed.

```
>cd <path>/<to>/<glyCompareCT>/<directory>
>./glyCompareCT
```

⚠ **CRITICAL:** If you use executables, please make sure you download the right version for your system (Linux, MacOS, or Windows).

Note: MacOS users might encounter the issue that the executable and the packages are disallowed to run due to the security settings of Apple. The error message is like "...because Apple cannot check it for malicious software". The users can run the following code to allow running glyCompareCT. This will not affect security settings for other software.

```
>xattr -r -d com.apple.quarantine <path>/<to>/<glyCompareCT>/<folder>/*
```

2. Next, set up the environment for Python scripts.
 - a. Python version after 3.8 is required (3.9 recommended) to run the python script. You need to install Conda to activate the Python running environment. A simple way to install both is to install Anaconda from here <https://docs.conda.io/projects/conda/en/latest/user-guide/install/index.html>.

- b. We store all required packages in the environment.yml file in the Github repo. Once conda is installed, you can enter the Github repo.

```
>cd /<path>/<to>/<glyCompareCT>/<github_repo>/
```

- c. Next, install required packages.

```
>conda env create -f environment.yml
```

- d. Finally, activate the conda environment with installed packages.

```
>conda activate glycompareCT
```

KEY RESOURCES TABLE

RESOURCE	SOURCE	IDENTIFIER
Deposited data		
Human milk oligosaccharide (HMO)	Mohammad et al. ²²	https://github.com/LewisLabUCSD/GlyCompare/tree/master/example_data/paper_hmo/source_data
Erythropoietin N-glycosylation	Čaval et al. ²⁰ and Yang et al. ²¹	https://github.com/LewisLabUCSD/GlyCompare/tree/master/example_data/paper_epo/source_data
Mucin-type O-glycans	Jin et al. ²⁴	https://github.com/LewisLabUCSD/GlyCompare/tree/master/example_data/Jin2017_Mucin/data-in
Mucin-type O-glycans	Adamczyk et al. ²³	https://github.com/LewisLabUCSD/GlyCompare/tree/master/example_data/Adamczyk_2018/source_data
Gangliosides	Sibille et al. ²⁵	https://github.com/LewisLabUCSD/GlyCompare/tree/master/example_data/Sibille2016_glycolipid/data-in
Composition data	Riley et al. ²⁶	https://github.com/LewisLabUCSD/GlyCompare/tree/master/example_data/Riley2019_SiteSpec/data-in
Software and algorithms		
GlyCompare	Bao et al. ¹	https://github.com/LewisLabUCSD/GlyCompare
Python	Version 3.9.9 or later	https://www.python.org/
Miniconda	Conda	https://docs.conda.io/en/latest/miniconda.html

STEP-BY-STEP METHOD DETAILS

Prepare data

⌚ Timing: < 30 min

With GlyCompareCT installed, users are ready to start processing your glycomics data, once the data files are formatted correctly.

1. Prepare two input files for GlyCompareCT: a glycan abundance file and a glycan annotation file, both in the csv (comma-separated values) format. For details of csv format, see <https://dev.socrata.com/docs/formats/csv.html>. You will format your data into these two file types as follow:
 - a. Construct the glycan abundance file with the absolute or relative abundance of glycans measured. It is a csv table where each entry in the table represents the abundance of the measured glycan in the sample of interest. (Figure 3A).

⚠ **CRITICAL:** Rows are samples and columns are glycans. The first row needs to be the glycan names, and the first column needs to be the sample names. See [troubleshooting 2](#) if needed.

A

	Glycan 1	Glycan 2	Glycan 3	Glycan 4
Sample 1	0.87	0.02	0.43	0.15
Sample 2	0.77	0.54	0.14	0.91
Sample 3	0.23	0.74	0.38	0.32
Sample 4	0.15	0.66	0.02	0.04

B

Name	Glycan Structure
Glycan 1	RES 1b:x-dglc-HEX-1:5 2b:b-dgal-HEX-1:5 3b:a-lgal-HEX-1:5 6:d LIN 1:1o(4+1)2d 2:2o(2+1)3d
Glycan 2	RES 1b:x-dglc-HEX-1:5 2b:a-lgal-HEX-1:5 6:d 3b:b-dgal-HEX-1:5 LIN 1:1o(3+1)2d 2:1o(4+1)3d
Glycan 3	RES 1b:x-dglc-HEX-1:5 2b:b-dgal-HEX-1:5 3b:b-dglc-HEX-1:5 4s:n-acetyl 5b:b-dgal-HEX-1:5 LIN 1:1o(4+1)2d 2:2o(3+1)3d 3:3d(2+1)4n 4:3o(4+1)5d
Glycan 4	RES 1b:b-dglc-HEX-1:5 2b:b-dgal-HEX-1:5 3b:a-dgro-dgal-NON-2: 6 1:al2:ketol3:d 4s:n-acetyl LIN 1:1o(4+1)2d 2:2o(3+2)3d 3:3d(5+1)4n

C

Name	Composition
Glycan 1	HexNAc(2)Hex(5)
Glycan 2	HexNAc(4)Hex(5)Fuc(1)
Glycan 3	HexNAc(4)Hex(5)Fuc(1)
Glycan 4	HexNAc(1)

Figure 3. Example input files

(A) Example glycan abundance table.

(B) Example Glycan annotation table with glycan names and glycan structure syntax in glycoCT format.

(C) Glycan annotation table with glycan names and glycan compositions.

- b. Construct the glycan annotation file (Figures 3B and 3C) with glycan names (column **Name**) and glycan structure (column **Glycan Structure**) or composition (column **Composition**). If glycan structures are not resolved, use glycan composition instead (Figure 3C).

Note: Accepted glycan structure syntax includes IUPAC-extended,²⁷ glycoCT,²⁸ WURCS,²⁹ GlyTouCan ID,³⁰ or Linear Code.³¹

Run GlyCompareCT

© Timing: < 20 min, on a dataset with 250 glycans and 17 samples using 2 CPUs.

Note: The codes below are of the syntax to run Python scripts. To run executables, you can simply replace “python glyCompareCT.py” with “./glyCompareCT”.

After formatting the input files, run GlyCompareCT in one of two modes depending on our data, whether it contains glycan structures or glycan composition data (common for glycoproteomics). The only difference between the input files is the inclusion of “Glycan Structure” or “Composition” columns in the glycan annotation file.

2. Glycan Structure mode. In this mode, you will specify glycan format as one of “iupac_extended”, “glycoCT”, “wurcs”, “glytoucan_id”, or “linear_code”. See example code to run GlycompareCT in structure mode, with “glycoCT” format. See [troubleshooting 2–5](#) if needed.

```
>python glyCompareCT.py structure -a abundance.csv \  
-v annotation.csv -o output_directory/ -p glycoCT
```

3. Composition mode. In this mode, you will format the glycan composition as multiples of monosaccharide with its occurrence followed in parenthesis (i.e., HexNAc(2)Fuc(1)). See example code to run GlycompareCT in composition mode. See [troubleshooting 5](#) if needed.

```
>python glyCompareCT.py composition -a abundance.csv \  
-v annotation.csv -o output_directory/
```

Note: Monosaccharide syntax in composition mode is relaxed. Theoretically GlyCompareCT can run any text of “XX(a)YY(b)”, where a and b are integers, in composition mode. So, it is important for the users to keep naming consistency in their glycoprofiles because any minor typo would be regarded as a new glycan composition.

Note: The “\” at the end of the line indicates the command has not finished; It includes the next line as well. Same to all code blocks below.

4. You only need one line of code to run GlyCompareCT; however, there are optional parameters you can specify in each step of the algorithm. The remaining steps describe the internal steps of GlyCompareCT to clarify parameters through the use of an example dataset of human gastric mucin glycoprofiles,²⁴ which contains 250 glycans across 17 samples, and has representative sparsity and correlation.

Note: GlyCompareCT can run on the example dataset in both structure and composition modes. The set of optional parameters for composition mode is a subset of parameters used for the structure mode. Thus, here we walk through the parameters in structure mode.

Note: GlyCompareCT wraps up all algorithm steps in a single package, which can be parameterized in the initial input, resulting in a single final output. The Parameter Tuning section explains the effects of optional parameters for each intermediate step.

Optional: While GlyCompareCT by default runs on 1 CPU, it supports parallelization across multiple CPUs. The users can add “-c < CPU number>” to specify CPU number. The following command will split computation across 16 CPUs. The runtime and memory usage with varying CPU numbers are tested on the human gastric mucin glycoprofiles²⁴ (250 glycans across 17 samples; [Figure 4](#)). Overall, the running memory (maximum resident set size) remains

consistent across different CPU numbers while runtime decreases as CPU number increases, with a diminishing time savings for larger CPU counts.

```
>python glyCompareCT.py structure -a abundance.csv \
-v annotation.csv -o output_directory/ -p glycoCT -c 16
```

Parameter tuning (optional)

⌚ Timing: < 20 min

GlyCompareCT supports optional parameters for each intermediate step of the algorithm. To clarify these, we describe each intermediate step in their respective subsections as follow.

5. Tune parameter for substructure identification and decomposition.
 - a. You can normalize input glycan abundance within each glycoprofile. You can specify two normalization methods: min-max or probabilistic quotient.³² The default command does not normalize input glycan abundance. To specify normalization, add “-n” followed by “min-max” or “prob-quot”.
 - i. Min-max normalizes each element x to $(x - \min) / (\max - \min)$, where min is the minimum value in the glycoprofile and max is the maximum value in the glycoprofile.
 - ii. The other normalization option is probabilistic quotient normalization.³² See Dieterle et al.³² for a detailed explanation of this method.

```
>python glyCompareCT.py structure -a abundance.csv -v annotation.csv -o output_directory/
-p glycoCT \ -n [min-max or prob-quot]
```

- b. For each glycan, GlyCompareCT breaks one or multiple glycosidic bonds to permute all possible substructures. You can leverage (“linkage-specific”) or ignore (“structural”) glycosidic linkage information during substructure extraction depending on the application.
 - i. The mode defaults to “linkage-specific”.
 - ii. You can specify the “structural” mode by adding “-s” to the command:

```
>python glyCompareCT.py structure -a abundance.csv \
-v annotation.csv -o output_directory/ -p glycoCT \
-s
```

- c. GlyCompareCT then sorts the substructure monosaccharide counts. You can annotate each substructure for either its occurrences or its binary existence in each glycan.
 - i. The default command uses occurrences (equals to “-m integer”).
 - ii. You can set binary existence instead of occurrences by adding “-m binary”:

```
>python glyCompareCT.py structure -a abundance.csv \
-v annotation.csv -o output_directory/ -p glycoCT \
-m [binary or integer]
```

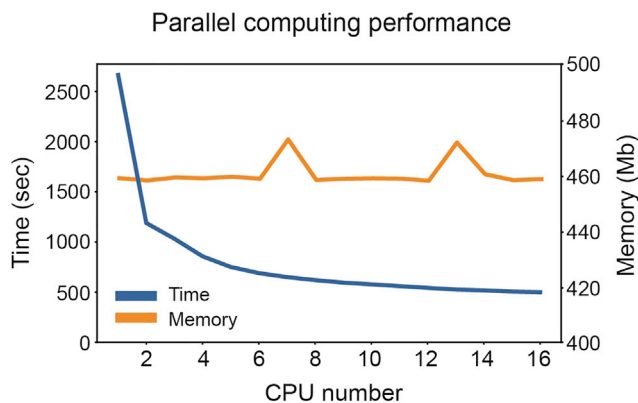


Figure 4. Runtime and memory change with CPU number

The time and memory usage for running the human gastric mucin glycoprofiles²⁴ (MO2) with varying CPU number. The test is run on Linux, specifically Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-188-generic x86_64).

- d. Next, GlyCompareCT builds a substructure abundance table from the glycan abundance table and substructure occurrences. Each substructure can appear in multiple glycans.

Note: The abundance of each substructure within the glycoprofile is calculated by summing over the products of each glycan abundance and the substructure occurrences in that glycan. Suppose the substructure occurrence of substructure i in glycan j is denoted as o_{ij} and the abundance of glycan j is a_j , then the abundance of substructure i in the glycoprofile denoted as s_i is $s_i = \sum_j o_{ij} a_j$. The substructure abundance values for each substructure across glycoprofiles represent the “decomposition” of glycoprofiles into substructure abundance profiles.

- i. The default setting normalizes the substructure abundance to the proportions in the glycoprofile.
- ii. You can output the substructure abundance table without normalization by adding “-b”:

```
>python glyCompareCT.py structure -a abundance.csv \
-v annotation.csv -o output_directory/ -p glycoCT \
-b
```

Note: GlyCompareCT identifies related substructures and quantifies substructure abundance correlation. Related substructures are retained only if they are not correlated. Specifically, when traversing the substructure network from a root, the smaller substructure of two perfectly correlated substructures is dropped. We call the minimal substructure set ‘glycomotifs’, and the corresponding subset of substructure abundances is the glycomotif abundance table.

Note: Glycan substructure relationships are represented in a directed acyclic substructure network where nodes are glycan substructures and edges directionally connect substructure nodes separated by the addition of one monosaccharide; therefore, edges represent known or hypothetical glycan synthesis reactions. Each node can have multiple parent nodes and multiple child nodes, since many glycans are substrate to multiple biosynthetic reactions and each yields distinct glycan products.

6. Tune parameters for glycomotif extraction from the substructure network.
 - a. Specify the substructure network root as either each monosaccharide (epitope mode) or a specific biosynthetic root. The default command is epitope mode, which does not specify one root but runs each monosaccharide as a root.

- b. Specify the biosynthetic root to focus the network reduction on biosynthetic or epitope motifs of interest. You can specify a provided root saved in GlyCompareCT, add “-r N” (N-glycan), “-r O” (O-glycan), or “-r lactose” (lactose):

```

>python glyCompareCT.py structure -a abundance.csv \
-v annotation.csv -o output_directory/ -p glycoCT \
-r [N or O or lactose]
  
```

Note: The N-glycan mode, “-r N”, specifies the root as “Man(b1-4)Glc2NAc(b1-4)Glc2NAc” in IUPAC; the O-glycan mode, “-r O”, specifies the root as “Gal2NAc” in IUPAC; the lactose mode, “-r lactose”, specifies the root as “Gal(b1-4)Glc”.

- c. Specify your own root of the substructure network. To do that, add “-r custom” and “-cr root_file” to the command, where root_file is a text file containing the root glycan in glycoCT format:

```

>python glyCompareCT.py structure -a abundance.csv \
-v annotation.csv -o output_directory/ -p glycoCT \
-r custom -cr root_file.txt
  
```

Note: Specifying “-r N” equals specifying the custom root to “RES\n1b:b-dglc-HEX-1:5\n2s:n-acetyl\n3b:b-dglc-HEX-1:5\n4s:n-acetyl\n5b:b-dman-HEX-1:5\nLIN\n1:1d(2 + 1)2\n2:1o(4 + 1)3d\n3:3d(2 + 1)4n\n4:3o(4 + 1)5d\n” in glycoCT format in the root_file.txt file. This structure equals ‘Mb4GNb4GNb’ in Linear Code format.

7. Annotate Glycomotifs. GlyCompareCT structure mode allows optional annotation and registration of output glycomotifs to GlyTouCan (<https://glytoucan.org/>). If set to annotate, GlyCompareCT will annotate any glycomotifs with existing GlyTouCan IDs and register the rest on GlyTouCan to retrieve the GlyTouCan IDs.
- The default command annotates the glycomotifs without GlyTouCan IDs, which is the same as adding “-e private”.
 - To annotate glycomotifs with GlyTouCan IDs while running GlyCompareCT in the structure mode, you need to add “-e register” and specify both your GlyTouCan contributor ID and API key.

```

>python glyCompareCT.py structure -a abundance.csv \
-v annotation.csv -o output_directory/ -p glycoCT \
-e register -C <contributor_ID> -A <API_key>
  
```

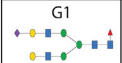


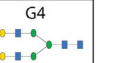
Note: GlyTouCan contributor ID and API key can be obtained upon signing up at GlyTouCan (<https://glytoucan.org/>). The information is at User menu - > Profile.

- c. You can also annotate glycomotifs with GlyTouCan IDs in the annotation mode. Once your glycans are registered, you can update the GlyTouCan IDs locally by running GlyCompareCT in annotation mode.

```

>python glyCompareCT.py annotate -n annotation_table.cs
  
```

A

					
Test 1	Samp 1	1	2	0	0
Test 2	Samp 1	1	2	1	0
Test 3	Samp 1	1	2	0	0
	Samp 2	0	1	0	0
Test 4	Samp 1	1	2	1	3

B

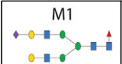

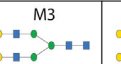

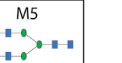
						
Test 1	Samp 1	1	3	0	0	0
Test 2	Samp 1	1	3	4	2	0
Test 3	Samp 1	1	3	3	1	0
	Samp 2	0	1	2	1	0
Test 4	Samp 1	1	6	7	2	3

Figure 5. Example input glycan abundance and output glycomotif abundance tables

(A) An example input glycan abundance table. G1-G4 are glycans. The values are occurrences of glycans in samples. (B) An example output glycomotif abundance table. M1-M5 are glycomotifs. The values are occurrences of glycomotifs in samples.

Note: When you run the structure mode with “-e register”, glycomotifs without existing GlyTouCan IDs will be registered on GlyTouCan. It may take a couple of days to complete the registration by GlyTouCan.

Note: The required annotation table can be the output glycomotif annotation csv file with suffix “_motif_annotation.csv” or a csv file with glycans in glycoCT format under “glycoCT” column or glycans in WURCS format under “WURCS” column.

EXPECTED OUTCOMES

The output of GlyCompareCT is a processed glycan substructure abundance table for all samples processed with the tool. The outputs will differ based on if the tool is run in Structure or Composition Mode.

In structure mode, the major output of GlyCompareCT is the glycomotif abundance table (with suffix “_motif_abd_table.csv”). Simple examples of an input glycan abundance table and its output glycomotif abundance table are shown in [Figure 5](#).

Just like the input glycan abundance table, the glycomotif abundance table has the column names as samples. Instead of glycans as rows in the glycan abundance table, the glycomotif abundance table has glycomotifs as rows. The row names correspond to the glycomotif reference index. The index can be matched to structures and GlyTouCan IDs (if “-e register” specified) in the annotation output file.

The GlyCompareCT output includes an annotation file with the glycomotif reference index, substructure index, WURCS, glycoCT, and GlyTouCan IDs (if “-e register” specified).

The output using the actual mucin dataset in the tutorial above²⁴ is shown here:

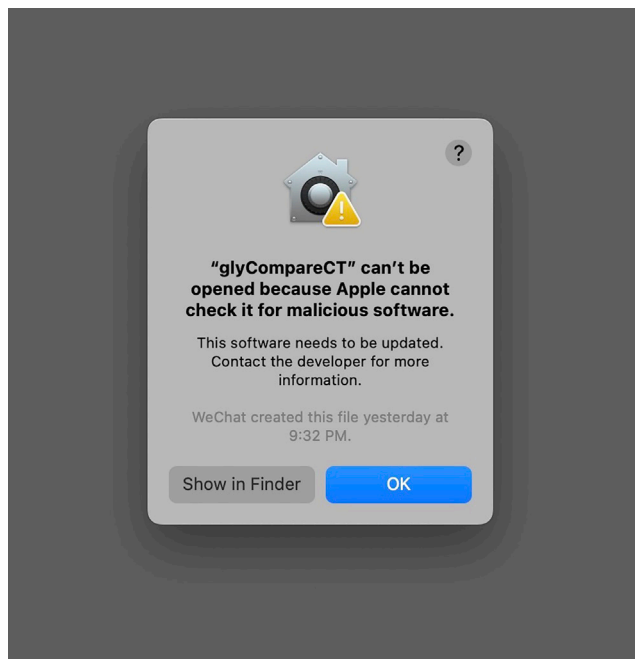


Figure 6. Example error message when running glyCompareCT executable on the MacOS

https://github.com/LewisLabUCSD/GlyCompareCT/blob/main/Examples/Jin2017/Jin2017_structure/Jin2017_output_data/Jin2017_motif_abd_table.csv.

In composition mode, the major output is a composition abundance table. This table is similar to the glycomotif abundance table but instead of glycomotif structures, each entry is a combination of monosaccharides occurrences. An example of such a combination from the dataset in the tutorial above is (('Hex', 1), ('HexNAC', 1), ('NeuAc', 2)).

The output using the actual mucin dataset in the tutorial above²⁴ is shown here: https://github.com/LewisLabUCSD/GlyCompareCT/blob/main/Examples/Jin2017/Jin2017_composition/Jin2017_compo_output_data/Jin2017_compo_motif_abd_table_composition.csv.

LIMITATIONS

A large portion of GlyCompareCT runtime is spent on glycomotif extraction from the substructure network. The runtime increases exponentially as the number of glycans increase. Therefore, we recommend using multiple CPUs if available, especially when the number of glycans in your dataset is large (> 100). Running GlyCompareCT with 2 CPUs on the human gastric mucin glycoprofiles,²⁴ which contains 250 glycans across 17 samples, completes in ~ 20 min using 460 Mb memory. If jobs exceed the desired runtime, we recommend cloud computing (e.g., Amazon Web Services or Google Cloud Platform) where CPU count can be easily increased.

If a custom root glycan is specified, it must be in glycoCT format.

TROUBLESHOOTING

Problem 1

GlyCompareCT executables on MacOS failed to run with popped-up error "...because Apple cannot check it for malicious software" (Figure 6), related to Environment Setup.

	Glycan 1	Glycan 2	Glycan 3	Glycan 4
Sample 1	0.1	0.2	0.3	0.4
Sample 2	0.1	0.2	0.3	0.4
Sample 3	0.1	0.2	0.3	0.4
Sample 4	0.1	0.2	0.3	0.4

	Sample 1	Sample 2	Sample 3	Sample 4
Glycan 1	0.1	0.1	0.1	0.1
Glycan 2	0.2	0.2	0.2	0.2
Glycan 3	0.3	0.3	0.3	0.3
Glycan 4	0.4	0.4	0.4	0.4

Figure 7. Correct and incorrect input glycan abundance table layouts

The input glycan abundance table in csv format should have the glycan names as columns and sample names as rows (left), not vice versa (right).

Potential solution

MacOS security settings disallows running software without Apple authorization. This issue can be solved by running the following command in the terminal. This command will not affect the security settings to other software.

```
>xattr -r -d com.apple.quarantine <path>/<to>/<glyCompareCT>/<folder>/*
```

Problem 2

GlyCompareCT in the structure mode failed to process the glycan abundance table because the table has mistakenly put glycans as rows and samples as columns (Figure 7), related to Prepare Data.

Potential solution

Manually swap the columns and rows, or open the csv file in Excel and follow their [documentation](#) on transposing data from rows to columns.

Problem 3

GlyCompareCT in the structure mode failed to process because the Name column of the glycan annotation table does not match the column of the glycan abundance table (Figure 8), related to step 2.

Potential solution

GlyCompareCT requires the exact match of glycans in the glycan annotation table and the glycan abundance table. So please make sure of it.

Problem 4

GlyCompareCT in the structure mode failed to process because there are unrecognized glycan structures syntax in the glycan annotation table (Figure 9), related to step 2.

```
Traceback (most recent call last):
  File "/Users/zhangyujie/Desktop/Lewis Lab/GlyCompareCT/glyCompareCT.py", line 685, in <module>
    main()
  File "/Users/zhangyujie/Desktop/Lewis Lab/GlyCompareCT/glyCompareCT.py", line 54, in main
    input_validation(args)
  File "/Users/zhangyujie/Desktop/Lewis Lab/GlyCompareCT/glyCompareCT.py", line 134, in input_validation
    assert set(glycan_abd.columns) == set(var_annot["Name"]), "The 'Name' column values of the variable annotation file are not the same as the column names of the glycan abundance table. Please check and correct it."
AssertionError: The 'Name' column values of the variable annotation file are not the same as the column names of the glycan abundance table. Please check and correct it.
```

Figure 8. Command line error message when glycan annotation Name column doesn't match glycan abundance table column names

```
Traceback (most recent call last):
  File "/Users/zhangyujie/Desktop/Lewis Lab/GlyCompareCT/glyCompareCT.py", line 685, in <module>
    main()
  File "/Users/zhangyujie/Desktop/Lewis Lab/GlyCompareCT/glyCompareCT.py", line 54, in main
    input_validation(args)
  File "/Users/zhangyujie/Desktop/Lewis Lab/GlyCompareCT/glyCompareCT.py", line 151, in input_validation
    assert not bad_glycan_names, ", ".join(bad_glycan_names) + "\n" + str(len(bad_glycan_names)) + " of " + str(var_a
nnot.shape[0]) + " glycans have invalid gstructure syntax. The glycan structures of the above names failed to be rec
ognized as " + str(args.data_syntax) + ". Non-recognized glycans are saved to bad_glycans.txt. Consider using -i to
ignore non-recognized glycans and proceed."
AssertionError: LNT
1 of 16 glycans have invalid gstructure syntax. The glycan structures of the above names failed to be recognized as
glycoCT. Non-recognized glycans are saved to bad_glycans.txt. Consider using -i to ignore non-recognized glycans and
proceed.
```

Figure 9. Command line error message when unrecognized glycan structure syntax exists

Potential solution

If in structure mode, please make sure the input glycan structures syntax (-P) is the same as the syntax in the Glycan Structure column of the annotation table; if in composition mode, please make sure the Composition column of the annotation table is of "XXX(a)YYY(b)" format, where "a" and "b" must be integers.

Or you can add "-i" to ignore the bad format glycans in both structure and composition mode.

Problem 5

GlyCompareCT in the structure / composition mode failed to process because there are negative values in the glycan abundance table (Figure 10), related to steps 2 and 3.

Potential solution

GlyCompareCT does not allow negative values in the glycan abundance table. It must be zero or positive.

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, Nathan E. Lewis (nlewisres@ucsd.edu).

Materials availability

This study did not generate new unique reagents.

Data and code availability

The dataset and code used during this study are available at <https://github.com/LewisLabUCSD/GlyCompareCT> and <https://github.com/LewisLabUCSD/GlyCompare>. The latest GlyCompareCT v1.2.1 release DOI is <https://doi.org/10.5281/zenodo.7511631>.

```
Traceback (most recent call last):
  File "/Users/zhangyujie/Desktop/Lewis Lab/GlyCompareCT/glyCompareCT.py", line 685, in <module>
    main()
  File "/Users/zhangyujie/Desktop/Lewis Lab/GlyCompareCT/glyCompareCT.py", line 54, in main
    input_validation(args)
  File "/Users/zhangyujie/Desktop/Lewis Lab/GlyCompareCT/glyCompareCT.py", line 122, in input_validation
    assert (glycan_abd < 0).sum().sum() == 0, "The input glycan abundance table can only have non-negative values. Negative v
alues detected. Please correct."
AssertionError: The input glycan abundance table can only have non-negative values. Negative values detected. Please correct.
```

Figure 10. Command line error message when negative values present in the glycan abundance table

ACKNOWLEDGMENTS

This work was conducted with generous support from NIGMS (R35 GM119850), NHLBI (K12HL 141956), and the Novo Nordisk Foundation through the National Biologics Facility at the Technical University of Denmark (NNF20SA0066621).

AUTHOR CONTRIBUTIONS

Software development, B.B., Y.Z., B.P.K.; writing, review, and editing, Y.Z., B.P.K., S.K., A.W.T.C., J.T.S., S.M.S., and N.E.L.; scientific evaluation, Y.Z., S.K., B.P.K.; funding acquisition, N.E.L.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

- Bao, B., Kellman, B.P., Chiang, A.W.T., Zhang, Y., Sorrentino, J.T., York, A.K., Mohammad, M.A., Haymond, M.W., Bode, L., and Lewis, N.E. (2021). Correcting for sparsity and interdependence in glycomics by accounting for glycan biosynthesis. *Nat. Commun.* **12**, 4988.
- Kellman, B.P., and Lewis, N.E. (2021). Big-data glycomics: tools to connect glycan biosynthesis to extracellular communication. *Trends Biochem. Sci.* **46**, 284–300.
- Reiding, K.R., Blank, D., Kuijper, D.M., Deelder, A.M., and Wuhrer, M. (2014). High-throughput profiling of protein N-glycosylation by MALDI-TOF-MS employing linkage-specific sialic acid esterification. *Anal. Chem.* **86**, 5784–5793.
- Wohlschlager, T., Scheffler, K., Forstenlehner, I.C., Skala, W., Senn, S., Damoc, E., Holzmann, J., and Huber, C.G. (2018). Native mass spectrometry combined with enzymatic dissection unravels glycoform heterogeneity of biopharmaceuticals. *Nat. Commun.* **9**, 1713.
- Black, A.P., Liang, H., West, C.A., Wang, M., Herrera, H.P., Haab, B.B., Angel, P.M., Drake, R.R., and Mehta, A.S. (2019). A novel mass spectrometry platform for multiplexed N-glycoprotein biomarker discovery from patient biofluids by antibody panel based N-glycan imaging. *Anal. Chem.* **91**, 8429–8435.
- Ashwood, C., Pratt, B., MacLean, B.X., Gundry, R.L., and Packer, N.H. (2019). Standardization of PGC-LC-MS-based glycomics for sample specific glycotyping. *Analyst* **144**, 3601–3612.
- Maxwell, E., Tan, Y., Tan, Y., Hu, H., Benson, G., Aizikov, K., Conley, S., Staples, G.O., Slysz, G.W., Smith, R.D., and Zaia, J. (2012). GlycReSoft: a software package for automated recognition of glycans from LC/MS data. *PLoS One* **7**, e45474.
- Hou, W., Qiu, Y., Hashimoto, N., Ching, W.-K., and Aoki-Kinoshita, K.F. (2016). A systematic framework to derive N-glycan biosynthesis process and the automated construction of glycosylation networks. *BMC Bioinf.* **17** (Suppl 7), 240.
- Kremkow, B.G., and Lee, K.H. (2018). Glyco-mapper: a Chinese hamster ovary (CHO) genome-specific glycosylation prediction tool. *Metab. Eng.* **47**, 134–142.
- Krambeck, F.J., Bennun, S.V., Andersen, M.R., and Betenbaugh, M.J. (2017). Model-based analysis of N-glycosylation in Chinese hamster ovary cells. *PLoS One* **12**, e0175376.
- Holst, S., van Pelt, G.W., Mesker, W.E., Tollenaar, R.A., Belo, A.I., van Die, I., Rombouts, Y., and Wuhrer, M. (2017). High-throughput and high-sensitivity mass spectrometry-based N-glycomics of mammalian cells. *Methods Mol. Biol.* **1503**, 185–196.
- Angel, P.M., Mehta, A., Norris-Caneda, K., and Drake, R.R. (2018). MALDI imaging mass spectrometry of N-glycans and tryptic peptides from the same formalin-fixed, paraffin-embedded tissue section. *Methods Mol. Biol.* **1788**, 225–241.
- Aoki-Kinoshita, K., Agravat, S., Aoki, N.P., Arpinar, S., Cummings, R.D., Fujita, A., Fujita, N., Hart, G.M., Haslam, S.M., Kawasaki, T., et al. (2016). GlyTouCan 1.0 – the international glycan structure repository. *Nucleic Acids Res.* **44**, D1237–D1242. <https://doi.org/10.1093/nar/gkv1041>.
- Campbell, M.P., Peterson, R., Mariethoz, J., Gasteiger, E., Akune, Y., Aoki-Kinoshita, K.F., Lisacek, F., and Packer, N.H. (2014). UniCarbKB: building a knowledge platform for glycoproteomics. *Nucleic Acids Res.* **42**, D215–D221.
- York, W.S., Mazumder, R., Ranzinger, R., Edwards, N., Kahsay, R., Aoki-Kinoshita, K.F., Campbell, M.P., Cummings, R.D., Feizi, T., Martin, M., et al. (2020). GlyGen: computational and informatics resources for glycoscience. *Glycobiology* **30**, 72–73. <https://doi.org/10.1093/glycob/cwz080>.
- Campbell, M.P., Nguyen-Khuong, T., Hayes, C.A., Flowers, S.A., Alagesan, K., Kolarich, D., Packer, N.H., and Karlsson, N.G. (2014). Validation of the curation pipeline of UniCarb-DB: building a global glycan reference MS/MS repository. *Biochim. Biophys. Acta* **1844**, 108–116.
- Alocchi, D., Mariethoz, J., Gastaldello, A., Gasteiger, E., Karlsson, N.G., Kolarich, D., Packer, N.H., and Lisacek, F. (2019). GlyConnect: glycoproteomics goes visual, interactive, and analytical. *J. Proteome Res.* **18**, 664–677.
- Aoki-Kinoshita, K.F., Lisacek, F., Mazumder, R., York, W.S., and Packer, N.H. (2020). The GlySpace Alliance: toward a collaborative global glycoinformatics community. *Glycobiology* **30**, 70–71. <https://doi.org/10.1093/glycob/cwz078>.
- Aoki-Kinoshita, K.F. (2021). Glycome informatics: using systems biology to gain mechanistic insights into glycan biosynthesis. *Curr. Opin. Chem. Eng.* **32**, 100683.
- Čaval, T., Tian, W., Yang, Z., Clausen, H., and Heck, A.J.R. (2018). Direct quality control of glycoengineered erythropoietin variants. *Nat. Commun.* **9**, 3342.
- Yang, Z., Wang, S., Halim, A., Schulz, M.A., Frodin, M., Rahman, S.H., Vester-Christensen, M.B., Behrens, C., Kristensen, C., Vakhrushev, S.Y., et al. (2015). Engineered CHO cells for production of diverse, homogeneous glycoproteins. *Nat. Biotechnol.* **33**, 842–844.
- Mohammad, M.A., Hadsell, D.L., and Haymond, M.W. (2012). Gene regulation of UDP-galactose synthesis and transport: potential rate-limiting processes in initiation of milk production in humans. *Am. J. Physiol. Endocrinol. Metab.* **303**, E365–E376.
- Adamczyk, B., Jin, C., Polom, K., Muñoz, P., Rojas-Macias, M.A., Zeeberg, D., Borén, M., Roviello, F., and Karlsson, N.G. (2018). Sample handling of gastric tissue and O-glycan alterations in paired gastric cancer and non-tumorigenic tissues. *Sci. Rep.* **8**, 242.
- Jin, C., Kenny, D.T., Skoog, E.C., Padra, M., Adamczyk, B., Vitzeva, V., Thorell, A., Venkatakrishnan, V., Lindén, S.K., and Karlsson, N.G. (2017). Structural diversity of human gastric mucin glycans. *Mol. Cell. Proteomics* **16**, 743–758.
- Sibille, E., Berdeaux, O., Martine, L., Bron, A.M., Creuzot-Garcher, C.P., He, Z., Thuret, G., Bretillon, L., and Masson, E.A.Y. (2016). Ganglioside profiling of the human retina: comparison with other ocular structures, brain and plasma reveals tissue specificities. *PLoS One* **11**, e0168794.

26. Riley, N.M., Hebert, A.S., Westphall, M.S., and Coon, J.J. (2019). Capturing site-specific heterogeneity with large-scale N-glycoproteome analysis. *Nat. Commun.* *10*, 1311.
27. International Union of Pure, Applied, Chemistry., Panico, R., Powell, W. H., and Richer, J.-C. (1993). Commission on the nomenclature of organic chemistry. *A Guide to IUPAC Nomenclature of Organic Compounds: Recommendations 1993* (Blackwell Science).
28. Herget, S., Ranzinger, R., Maass, K., and Lieth, C.-W.V.D. (2008). GlycoCT—a unifying sequence format for carbohydrates. *Carbohydr. Res.* *343*, 2162–2171.
29. Tanaka, K., Aoki-Kinoshita, K.F., Kotera, M., Sawaki, H., Tsuchiya, S., Fujita, N., Shikanai, T., Kato, M., Kawano, S., Yamada, I., and Narimatsu, H. (2014). WURCS: the Web3 unique representation of carbohydrate structures. *J. Chem. Inf. Model.* *54*, 1558–1566. <https://doi.org/10.1021/ci400571e>.
30. Tiemeyer, M., Aoki, K., Paulson, J., Cummings, R.D., York, W.S., Karlsson, N.G., Lisacek, F., Packer, N.H., Campbell, M.P., Aoki, N.P., et al. (2017). GlyTouCan: an accessible glycan structure repository. *Glycobiology* *27*, 915–919.
31. Banin, E., Neuberger, Y., Altshuler, Y., Halevi, A., Inbar, O., Nir, D., and Dukler, A. (2002). A novel linear code nomenclature for complex carbohydrates. *Trends Glycosci. Glycotechnol.* *14*, 127–137. <https://doi.org/10.4052/tigg.14.127>.
32. Dieterle, F., Ross, A., Schlotterbeck, G., and Senn, H. (2006). Probabilistic quotient normalization as robust method to account for dilution of complex biological mixtures. Application in 1H NMR metabolomics. *Anal. Chem.* *78*, 4281–4290.