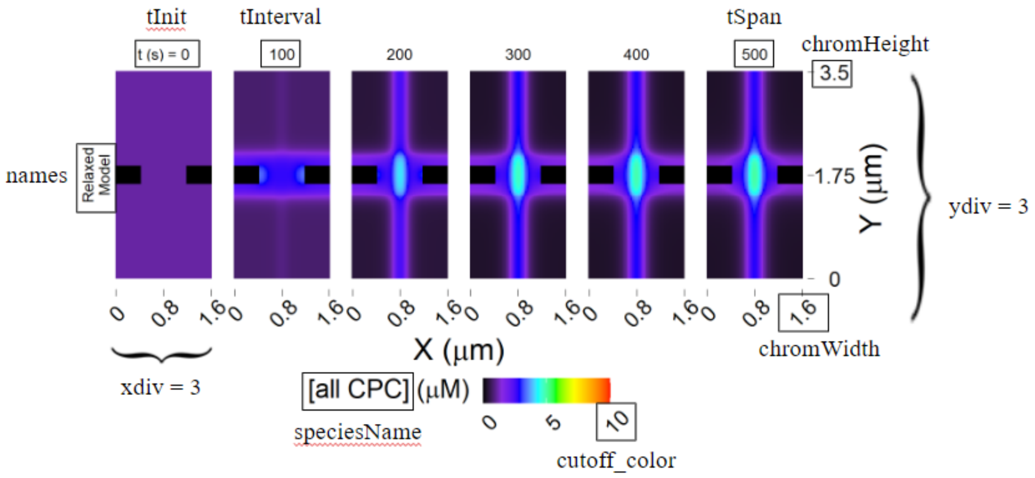
“vcell_heatmap.R”

Description:

This function generates heat maps based on concentration data from VCell simulations for specific species and time intervals. The function reads simulation data from CSV files, processes the data, calculates concentrations, and plots the results using the “ggplot” package. The resulting plot is saved as a PNG file.

Arguments:

- **“SimID”**: A vector of strings containing SimIDs
- **“names”**: A string specifying the type of model
- **“species”**: A string specifying the species of interest
- **“speciesName”**: A string for saving and display within the plot
- **“cutoff_color”**: The maximum concentration value to be displayed on the color scale
- **“tInit”**: The initial time in seconds
- **“tSpan”**: The time span in seconds
- **“tInterval”**: The time interval in seconds
- **“desiredInterval”**: The desired interval in seconds for data spacing
- **“chromWidth”**: The width of the chromosome in um
- **“chromHeight”**: The height of the chromosome in um
- **“dataDim”**: The number of rows and columns in imported data (y, x)
- **“row_1”**: The starting row for concentration data
- **“row_2”**: The ending row for concentration data
- **“col_1”**: The starting column for concentration data
- **“col_2”**: The ending column for concentration data
- **“xdiv”**: The number of divisions desired on the x-axis of the output plot
- **“ydiv”**: The number of divisions desired on the y-axis of the output plot
- **“importPath”**: The path where the data files are located
- **“exportPath”**: The path where the generated heatmap will be saved



“all_plot.R”

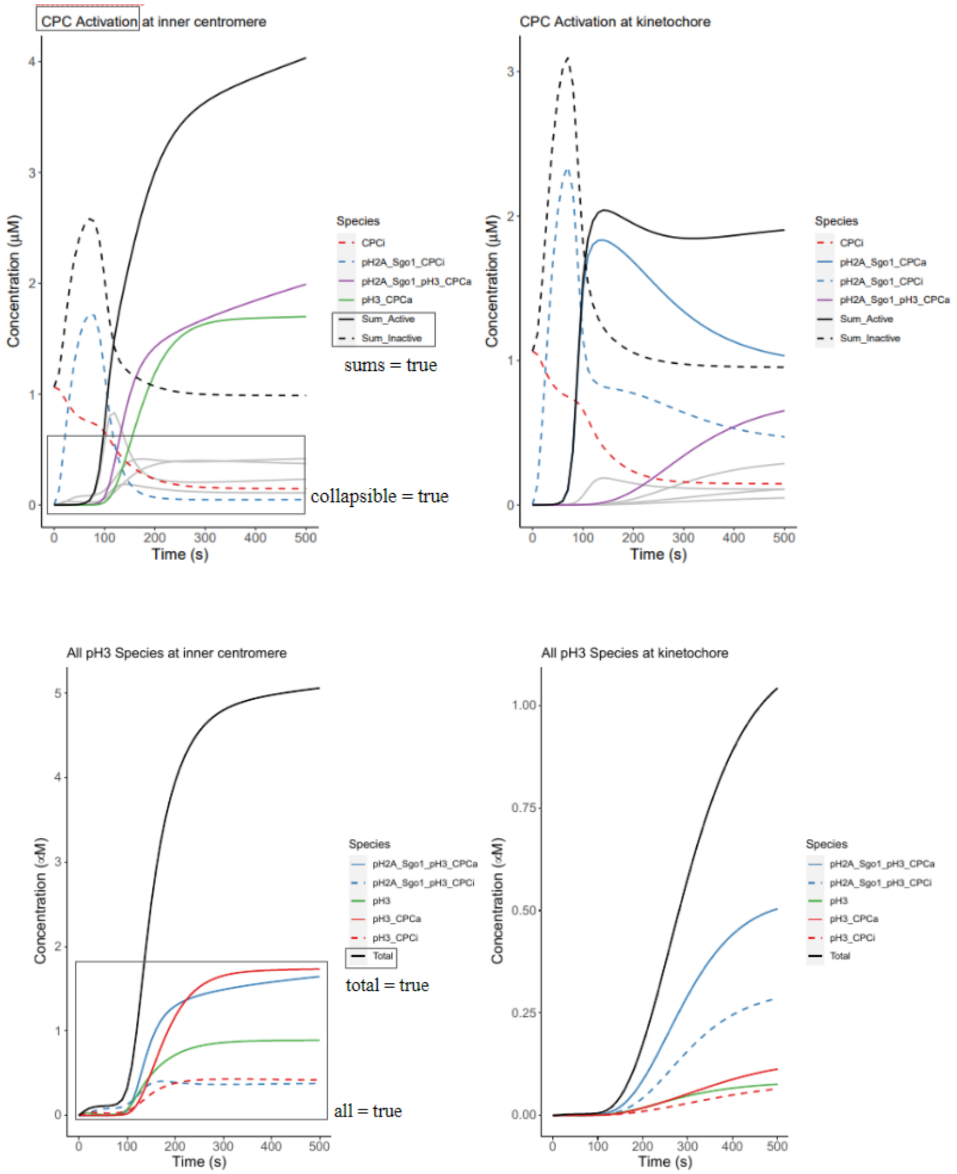
Description:

This function generates line graphs based on concentration data from VCell simulations for specific species and time intervals. This function uses the same basis as “vcell_heatmaps.R” to read simulation data, process it, and plot it. Two regions are plotted in one plot: the inner centromere and the kinetochores. The mean of the concentrations inside each region is calculated and plotted against time. The results are saved as a pdf and a png file (which is saved with the same name as a pdf + _1, ex. CPC_plot_1).

Arguments:

- **“SimID”**: A vector of strings containing SimIDs
- **“names”**: A string specifying the type of model
- **“all_data”**: A list of lists specifying all species within a single line plot
- **“all_species”**: A list of all the species
- **“species_info_list”**: A list of lists specifying naming details of each line plot
    - identity = species_info_list[[species]][1]
        - String
        - Specifies the name to be used for saving, ex. **CPC**_plot
    - speciesInactive = species_info_list[[species]][2]
        - String
        - Specifies the name of inactive species used in **inactive plots**
    - speciesActive = species_info_list[[species]][3]
        - String
        - Specifies the name of active species used in **active plots**
    - speciesFull = species_info_list[[species]][4]
        - String
        - Specifies the name of **both active and inactive** species used in plots
    - sums = species_info_list[[species]][5]
        - Boolean
        - Specifies whether sums of inactive and active species should be added
            - Active: Black, Solid
            - Inactive: Black, Dashed
    - total = species_info_list[[species]][6]
        - Boolean
        - Specifies whether the sum of all species should be added
            - Active: Black, Solid
    - full = species_info_list[[species]][7]
        - Boolean
        - Specifies whether all species should be added to line plots
    - collapsible = species_info_list[[species]][8]
        - Boolean
        - Specifies whether only the top 4 species and their sums/total should be specified

- **“tInit”**: The initial time in seconds
- **“tSpan”**: The time span in seconds
- **“chromWidth”**: The width of the chromosome in um
- **“chromHeight”**: The height of the chromosome in um
- **“dataDim”**: The number of rows and columns in imported data (y, x)
- **“row_1”**: The starting row for concentration data
- **“row_2”**: The ending row for concentration data
- **“col_1”**: The starting column for concentration data
- **“col_2”**: The ending column for concentration data
- **“importPath”**: The path where the data files are located
- **“exportPath”**: The path where the generated line plots will be saved
- **“linewidth”**: The width of lines used in the line plots



“save_plots.R”

Description:

This function is designed to generate and save plots, including heatmaps and line plots, based on the provided simulation data. It utilizes two custom functions, **‘vcell_heatmap’** and **‘all_plot’** to create visualizations and save them to specified export paths.

Arguments:

- **“SimID”**: A vector of strings containing SimIDs
- **“names”**: A string specifying the type of model
- **“heatmap_species”**: A list of lists specifying the species within each heatmap
- **“heatmap_info_list”**: A list of lists specifying the names of each heatmap
- **“species_info_list”**: A list of lists specifying naming details of each line plot
  - See **‘all_plots.R’** documentation for more details

- “**tInit**”: The initial time in seconds
- **“tSpan”**: The time span in seconds
- **“desiredInterval”**: The desired interval in seconds for data spacing
- **“cutoff”**: The maximum value to be displayed on the heatmap
- **“funcPath”**: The path to VCell analysis functions
- “**importPath**”: The path where the data files are located
- **“exportPath”**: The path where the generated line plots will be saved

“vcell_table.R”

Description:

This function is designed to output a table of the difference between the inner centromere and kinetochore values at specific time points. The function accepts a species vector and takes the sum of all the species as a bulk concentration at certain points.

THIS FUNCTION CAN ACCEPT MULTIPLE SIMULATIONS AT ONCE AS LONG AS YOU PROVIDE THE CORRESPONDING ‘VAR’ DESCRIPTIONS IN THE SAME ORDER.

Arguments:

- **“sims”**: A vector of strings containing SimIDs
- **“vars”**: A vector of strings containing the description of corresponding sims
- **“tPoints”**: A list of time points to include in the table
- **“all_species”**: A list of species to sum up
- **“name”**: A string for naming the table within the plot directory
- **“chromWidth”**: The width of the chromosome in um
- **“chromHeight”**: The height of the chromosome in um
- **“dataDim”**: The number of rows and columns in imported data (y, x)
- **“row_1”**: The starting row for concentration data
- **“row_2”**: The ending row for concentration data
- **“col_1”**: The starting column for concentration data
- **“col_2”**: The ending column for concentration data
- “**importPath**”: The path where the data files are located
- **“exportPath”**: The path where the generated line plots will be saved

“vcell_run_v3_mail.R”

Dependencies:

ggplot2, gridExtra, purrr, latex2exp, stringr, lemon, utils, tictoc, tidyverse, tibble, scales, xlsx, pdftools, rgoogleslides

Installing rgoogleslides:

```
install.packages("devtools")
library(devtools)
devtools::install_github("hairizuanbinnoorazman/rgoogleslides", build_vignettes = TRUE)
```

Setting up:

Google Cloud Platform > APIs & Services > Library > Search “Google Drive API” > Turn on
> Search “Google Slides API” > Turn on
> Search “Cloud Storage” > Turn on

Google Cloud Platform > APIs & Services > Credentials > Create Credentials > OAuth client ID > Application type = Desktop app > Copy **Client ID** and **Client Secret**

Google Cloud Platform > Cloud Storage > Buckets > Create bucket > Name bucket > Create > Enforce public access prevention on this bucket > **Get Name of Bucket**

Google Cloud Platform > Select or Create a Project > IAM & Admin > Service Accounts > Create Service Account > Name / Description > Select a Role > Storage Object Admin > Continue > Actions > Manage Keys > Add Key > Create New Key > JSON > **Get Path to JSON Key**

Description:

This is the main script used for VCell quantitative analysis. After installing needed packages, function, import, and desktop paths are specified. All functions are updated according to their last saved versions. Number of heatmaps required is stored in **‘H’** and the number of line plots required is stored in **‘L’**. Data to be changed is marked with a commented **#Change**. Specified species can be used or novel ones can be added for further use.

Once all changes have been made, the script creates a new directory by the specific simulation ID’s corresponding **‘var’** in the specified **‘exportPath’** and generates all plots in that directory. By default, a table with a single row is added to the plot directory.

A new **‘sims’** and **‘var’** specify what simulations to add to a Google slideshow. Complete authorization as follows:

```
Sys.setenv(“GCS_DEFAULT_BUCKET” = “NAME_OF_YOUR_BUCKET”)
Sys.setenv("GCS_AUTH_FILE"="PATH_TO_JSON_KEY")
```

```
gcs_global_bucket("NAME_OF_YOUR_BUCKET")
authorize("CLIENT_ID",
        "CLIENT_SECRET")

gcs_auth("PATH_TO_JSON_BUCKET")
```

'slide_plots' will house what plots you want to add to the presentation, given that they exist within the simulation plot folder.

If you want to append simulations to the end of a presentation, obtain the slide_id.

For example:
https://docs.google.com/presentation/d/**1C4GAaA6RdyQU2zMXqHwUa3rS1zyECDjaqNd_i4s1oPA**/edit#slide=id.g25dce96db8f_0_0

The bolded part is the slide_id.
slide_id ← **1C4GAaA6RdyQU2zMXqHwUa3rS1zyECDjaqNd_i4s1oPA**

Otherwise, create a new slideshow with:

```
slide_id <- rgoogleslides::create_slides(title)
```

Important Variables:

"funcpath":	Path to functions
"importPath":	Path to simulation data
"exportPath":	Path to plot folder destination
"dataDim":	Dimension of chromosome
"H":	Number of heat map plots
"L":	Number of line graph plots
"names":	Name of model type (ex. "Relaxed Model")