# Case Study

Using the cohort of patients who were diagnosed with Stage IV adenocarcinoma NSCLC, we will:

1. **Process data** into a analysis-ready matrix of gene alteration events

2. **Summarize genomic alteration frequencies** and analyze differences between males and females

```r
nsclc_cohort <- create_analytic_cohort(
  data_synapse =
    nsclc_synapse_data$NSCLC_v2.0,
  stage_dx = c("Stage IV"),
  histology = "Adenocarcinoma",
  regimen_drugs =
    c("Carboplatin, Pemetrexed Disodium",
      "Cisplatin, Pemetrexed Disodium",
      "Bevacizumab, Carboplatin, Pemetrexed Disodium",
      "Bevacizumab, Cisplatin, Pemetrexed Disodium"),
  regimen_type = "Exact",
  regimen_order = 1,
  regimen_order_type = "within cancer",
  return_summary = TRUE
)
```

| Characteristic | N = 241[1] |
|---|---|
| **naaccr_sex_code** | |
| Female | 145 (60%) |
| Male | 96 (40%) |

[1] n (%)

# Overview of Genomic Data

We will be processing and analyzing data on:

## 1) Mutations

```
1  mutations <- nsclc_synapse_data$NSCLC_v2.0$mutations_extended
```

## 2) Discrete Copy Number Alterations

```
1  cna <- nsclc_synapse_data$NSCLC_v2.0$cna
```

## 3) Fusions

```
1  fusions <- nsclc_synapse_data$NSCLC_v2.0$fusions
```

# Processing Data

# Issues When Processing Multi-Institutional Genomic Data

# Issues When Processing Multi-Institutional Genomic Data

*i*    1. **Multiple Samples Per Patient**

*i*

*i*

*i*

# Issues When Processing Multi-Institutional Genomic Data

ⓘ 1. **Multiple Samples Per Patient**

- We often need to select a single sample per patient for analyses

ⓘ

ⓘ

ⓘ

# Issues When Processing Multi-Institutional Genomic Data

1. **Multiple Samples Per Patient**
   - We often need to select a single sample per patient for analyses

2. **Data Formats & Gene Standards Often Inconsistent**

# Issues When Processing Multi-Institutional Genomic Data

ⓘ 1. **Multiple Samples Per Patient**

- We often need to select a single sample per patient for analyses

ⓘ 2. **Data Formats & Gene Standards Often Inconsistent**

- Column names, data formats and gene names may differ between studies or even within studies

ⓘ

ⓘ

# Issues When Processing Multi-Institutional Genomic Data

(i) 1. **Multiple Samples Per Patient**

- We often need to select a single sample per patient for analyses

(i) 2. **Data Formats & Gene Standards Often Inconsistent**

- Column names, data formats and gene names may differ between studies or even within studies

(i) 3. **Cohort Inclusion**

(i)

# Issues When Processing Multi-Institutional Genomic Data

1. **Multiple Samples Per Patient**
   - We often need to select a single sample per patient for analyses

2. **Data Formats & Gene Standards Often Inconsistent**
   - Column names, data formats and gene names may differ between studies or even within studies

3. **Cohort Inclusion**
   - Samples with no alterations may be dropped when pulling data

# Issues When Processing Multi-Institutional Genomic Data

ⓘ 1. **Multiple Samples Per Patient**
- We often need to select a single sample per patient for analyses

ⓘ 2. **Data Formats & Gene Standards Often Inconsistent**
- Column names, data formats and gene names may differ between studies or even within studies

ⓘ 3. **Cohort Inclusion**
- Samples with no alterations may be dropped when pulling data

ⓘ 4. **Multi-Institutional Studies Use Several Gene Panels**

# Issues When Processing Multi-Institutional Genomic Data

**1. Multiple Samples Per Patient**

- We often need to select a single sample per patient for analyses

**2. Data Formats & Gene Standards Often Inconsistent**

- Column names, data formats and gene names may differ between studies or even within studies

**3. Cohort Inclusion**

- Samples with no alterations may be dropped when pulling data

**4. Multi-Institutional Studies Use Several Gene Panels**

- Samples may be sequenced using different panels, therefore the non-overlapping genes have to be annotated as missing

# Select One Sample Per Patient

> ⓘ 1. **Multiple Samples Per Patient**
>
>   - We often need to select a single sample per patient for analyses

| Argument | Description |
|---|---|
| data_cohort | Output object of the create_analytic_cohort function. |
| oncotree_code | Character vector specifying which sample OncoTree codes to keep. See 'cpt_oncotree_code' column of data_cohort. |
| sample_type | Character specifying which type of genomic sample to prioritize. Options are 'Primary', 'Local', and 'Metastasis'. Default is to not select a NGS sample based on the sample type. |
| min_max_time | Character specifying if the first or last genomic sample recorded should be kept. Options are 'min' (first) and 'max' (last). |

# Select One Sample Per Patient

> (i) 1. **Multiple Samples Per Patient**
>
> - We often need to select a single sample per patient for analyses

- Some patients have multiple sequencing reports (e.g. pre & post treatment samples, primary and metastatic samples)

| Argument | Description |
| --- | --- |
| data_cohort | Output object of the create_analytic_cohort function. |
| oncotree_code | Character vector specifying which sample OncoTree codes to keep. See 'cpt_oncotree_code' column of data_cohort. |
| sample_type | Character specifying which type of genomic sample to prioritize. Options are 'Primary', 'Local', and 'Metastasis'. Default is to not select a NGS sample based on the sample type. |
| min_max_time | Character specifying if the first or last genomic sample recorded should be kept. Options are 'min' (first) and 'max' (last). |

# Select One Sample Per Patient

> (i) 1. **Multiple Samples Per Patient**
>
>   • We often need to select a single sample per patient for analyses

- Some patients have multiple sequencing reports (e.g. pre & post treatment samples, primary and metastatic samples)

- Two approaches to analysis:

| Argument | Description |
|---|---|
| data_cohort | Output object of the create_analytic_cohort function. |
| oncotree_code | Character vector specifying which sample OncoTree codes to keep. See 'cpt_oncotree_code' column of data_cohort. |
| sample_type | Character specifying which type of genomic sample to prioritize. Options are 'Primary', 'Local', and 'Metastasis'. Default is to not select a NGS sample based on the sample type. |
| min_max_time | Character specifying if the first or last genomic sample recorded should be kept. Options are 'min' (first) and 'max' (last). |

# Select One Sample Per Patient

> ⓘ 1. **Multiple Samples Per Patient**
>
>   • We often need to select a single sample per patient for analyses

• Some patients have multiple sequencing reports (e.g. pre & post treatment samples, primary and metastatic samples)

• Two approaches to analysis:

   1. Aggregate information across sequencing reports (e.g. "ever" altered)

| Argument | Description |
| --- | --- |
| data_cohort | Output object of the create_analytic_cohort function. |
| oncotree_code | Character vector specifying which sample OncoTree codes to keep. See 'cpt_oncotree_code' column of data_cohort. |
| sample_type | Character specifying which type of genomic sample to prioritize. Options are 'Primary', 'Local', and 'Metastasis'. Default is to not select a NGS sample based on the sample type. |
| min_max_time | Character specifying if the first or last genomic sample recorded should be kept. Options are 'min' (first) and 'max' (last). |

# Select One Sample Per Patient

> (i) 1. **Multiple Samples Per Patient**
>
> - We often need to select a single sample per patient for analyses

- Some patients have multiple sequencing reports (e.g. pre & post treatment samples, primary and metastatic samples)

- Two approaches to analysis:

  1. Aggregate information across sequencing reports (e.g. "ever" altered)

  2. Select one unique sequencing report -> **We focus on this approach**

| Argument | Description |
|---|---|
| data_cohort | Output object of the create_analytic_cohort function. |
| oncotree_code | Character vector specifying which sample OncoTree codes to keep. See 'cpt_oncotree_code' column of data_cohort. |
| sample_type | Character specifying which type of genomic sample to prioritize. Options are 'Primary', 'Local', and 'Metastasis'. Default is to not select a NGS sample based on the sample type. |
| min_max_time | Character specifying if the first or last genomic sample recorded should be kept. Options are 'min' (first) and 'max' (last). |

# Select One Sample Per Patient

> (i) 1. **Multiple Samples Per Patient**
>
>   • We often need to select a single sample per patient for analyses

- Some patients have multiple sequencing reports (e.g. pre & post treatment samples, primary and metastatic samples)

- Two approaches to analysis:

  1. Aggregate information across sequencing reports (e.g. "ever" altered)

  2. Select one unique sequencing report -> **We focus on this approach**

- We can use `select_unique_ngs()` to select 1 sample per patient

| Argument | Description |
|---|---|
| data_cohort | Output object of the create_analytic_cohort function. |
| oncotree_code | Character vector specifying which sample OncoTree codes to keep. See 'cpt_oncotree_code' column of data_cohort. |
| sample_type | Character specifying which type of genomic sample to prioritize. Options are 'Primary', 'Local', and 'Metastasis'. Default is to not select a NGS sample based on the sample type. |
| min_max_time | Character specifying if the first or last genomic sample recorded should be kept. Options are 'min' (first) and 'max' (last). |

# Select One Sample Per Patient

> ⓘ 1. **Multiple Samples Per Patient**
>
>   - We often need to select a single sample per patient for analyses

```
1  nrow(nsclc_cohort$cohort_ngs)
```

[1] 262

```
1  nsclc_samp <- select_unique_ngs(
2    data_cohort = nsclc_cohort$cohort_ngs,
3    oncotree_code = "LUAD",
4    sample_type = "Metastasis",
5    min_max_time = "max"
6  )
7
8  nrow(nsclc_samp)
```

[1] 241

# Format Data in Analysis-ready Matrix

| sample_id | CREBBP | GLI2 | KRAS | MAP3K1 | PIK3C2B | PBRM1 | MYC.Del |
|---|---|---|---|---|---|---|---|
| GENIE-DFCI-004022-1313 | 0 | 0 | 0 | 0 | 0 | NA | 0 |
| GENIE-DFCI-000013-8840 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000136-6004 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000215-8010 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000381-9526 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# Format Data in Analysis-ready Matrix

- Next we want to get our genomic data in an **analysis-friendly format**.

| sample_id | CREBBP | GLI2 | KRAS | MAP3K1 | PIK3C2B | PBRM1 | MYC.Del |
|---|---|---|---|---|---|---|---|
| GENIE-DFCI-004022-1313 | 0 | 0 | 0 | 0 | 0 | NA | 0 |
| GENIE-DFCI-000013-8840 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000136-6004 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000215-8010 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000381-9526 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# Format Data in Analysis-ready Matrix

- Next we want to get our genomic data in an **analysis-friendly format**.

- `create_gene_binary()` from {gnomeR} will give us a data frame of *n* **patients x** *p* **alterations**.

| sample_id | CREBBP | GLI2 | KRAS | MAP3K1 | PIK3C2B | PBRM1 | MYC.Del |
|-----------|--------|------|------|--------|---------|-------|---------|
| GENIE-DFCI-004022-1313 | 0 | 0 | 0 | 0 | 0 | NA | 0 |
| GENIE-DFCI-000013-8840 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000136-6004 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000215-8010 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000381-9526 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# Format Data in Analysis-ready Matrix

- Next we want to get our genomic data in an **analysis-friendly format**.

- `create_gene_binary()` from {gnomeR} will give us a data frame of *n* **patients** x *p* **alterations**.

- Alteration columns are denoted by the **gene name** if mutation (e.g. TP53) or **gene name + .Amp, .fus, .Del** (`TP53.Del`) for other alterations types.

| sample_id | CREBBP | GLI2 | KRAS | MAP3K1 | PIK3C2B | PBRM1 | MYC.Del |
|---|---|---|---|---|---|---|---|
| GENIE-DFCI-004022-1313 | 0 | 0 | 0 | 0 | 0 | NA | 0 |
| GENIE-DFCI-000013-8840 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000136-6004 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000215-8010 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000381-9526 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# Format Data in Analysis-ready Matrix

- Next we want to get our genomic data in an **analysis-friendly format**.

- `create_gene_binary()` from {gnomeR} will give us a data frame of *n* **patients** x *p* **alterations**.

- Alteration columns are denoted by the **gene name** if mutation (e.g. TP53) or **gene name + .Amp, .fus, .Del** (`TP53.Del`) for other alterations types.

- Each cell will have 0 if no alteration, 1 if altered, or NA if that gene was not tested in that patient.

| sample_id | CREBBP | GLI2 | KRAS | MAP3K1 | PIK3C2B | PBRM1 | MYC.Del |
|-----------|--------|------|------|--------|---------|-------|---------|
| GENIE-DFCI-004022-1313 | 0 | 0 | 0 | 0 | 0 | NA | 0 |
| GENIE-DFCI-000013-8840 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000136-6004 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000215-8010 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000381-9526 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# Get Data in Standardized Format

> ⓘ  2. **Data Formats & Gene Standards Often Inconsistent**
>
>    • Column names, data formats and gene names may differ between studies or even within studies!

**{gnomeR}** functions are designed to work with standard alteration data formats used in common platforms like cBioPortal.

GENIE CNA and fusion data are slightly different than this standard, therefore we need to reformat it using:

- `gnomeR::reformat_fusions()`
- `gnomeR::pivot_cna_longer()`

Documentation on data formats: https://docs.cbioportal.org/file-formats/

# Get Data in Standardized Format

Format fusions to follow the cBioPortal standard:

## Before:

```
1  fusions
```

| Hugo_Symbol | Tumor_Sample_Barcode | Fusion |
|---|---|---|
| TP53 | GENIE-MSK-P-0004827-T01-IM5 | TP53-intragenic |
| XPO1 | GENIE-MSK-P-0004827-T01-IM5 | XPO1-USP34 fusion |
| USP34 | GENIE-MSK-P-0004827-T01-IM5 | XPO1-USP34 fusion |

## After:

```
1  reformat_fusion(fusions)
```

| sample_id | site_1_hugo_symbol | site_2_hugo_symbol | fusion |
|---|---|---|---|
| GENIE-MSK-P-0004827-T01-IM5 | TP53 | NA | TP53 |
| GENIE-MSK-P-0004827-T01-IM5 | USP34 | XPO1 | USP34-XPO1 |

# Get Data in Standardized Format

Format CNA to follow the cBioPortal standard:

## Before:

```
1  cna
```

| Hugo_Symbol | GENIE.DFCI.003908.234520 | GENIE.DFCI.002183.6917 |
|---|---|---|
| APC | 1 | 0 |
| ARID1B | -1 | 0 |
| BCL2 | -1 | 0 |
| TP53 | 0 | 0 |

## After:

```
1  gnomeR::pivot_cna_longer(cna)
```

| hugo_symbol | sample_id | alteration |
|---|---|---|
| APC | GENIE-DFCI-003908-234520 | gain |
| ARID1B | GENIE-DFCI-003908-234520 | loss |
| BCL2 | GENIE-DFCI-003908-234520 | loss |
| BCL6 | GENIE-DFCI-002183-6917 | gain |
| CARD11 | GENIE-DFCI-002183-6917 | gain |

Documentation on data formats: https://docs.cbioportal.org/file-formats/

# Format Data in Analysis-ready Matrix

## Basic code:

```
1  gnomeR::create_gene_binary(
2    mutation = mutations,
3    cna = reformat_cna,
4    fusion = reformat_fusions)
```

# Format Data in Analysis-ready Matrix

- Data is now in standardized format and can be processed using {gnomeR} create_gene_binary()

**Basic code:**

```
1  gnomeR::create_gene_binary(
2    mutation = mutations,
3    cna = reformat_cna,
4    fusion = reformat_fusions)
```

# Format Data in Analysis-ready Matrix

- Data is now in standardized format and can be processed using {gnomeR} `create_gene_binary()`

- We will add additional arguments to `create_gene_binary()` to help address remaining data processing issues

**Basic code:**

```
1  gnomeR::create_gene_binary(
2    mutation = mutations,
3    cna = reformat_cna,
4    fusion = reformat_fusions)
```

# Cohort Inclusion

```
1  gnomeR::create_gene_binary(
2    mutation = mutations,
3    cna = reformat_cna,
4    fusion = reformat_fusions
5    samples = nsclc_samp$cpt_genie_sample_id)
```

The `samples` argument will ensure all study IDs have a row in resulting analysis data, even if they are not present in genomic files

# Not All Patients Tested on Same Panel

> ⓘ  4. **Multi-Institutional Studies Use Several Gene Panels**
>
>    - Samples may be sequenced using different panels therefore the non overlapping genes have to be annotated as missing

```r
gnomeR::create_gene_binary(
  mutation = mutations,
  cna = reformat_cna,
  fusion = reformat_fusions,
  samples = nsclc_samp$cpt_genie_sample_id,
  specify_panel = nsclc_panels)
```

The `specify_panels` argument can insert NAs when we know that gene was not tested for a specific set of patients.

# Not All Patients Tested on Same Panel

To use `specify_panels`, we first need to create a data frame indicating which patient IDs were sequenced on which panels.

```
1  nsclc_panels <- data.frame(
2    sample_id = nsclc_samp$cpt_genie_sample_id,
3    panel_id = nsclc_samp$cpt_seq_assay_id) %>%
4    mutate(panel_id = ifelse(!is.na(panel_id),
5                            panel_id, "no"))
6
7  nsclc_panels %>% head()
```

| sample_id | panel_id |
|---|---|
| GENIE-DFCI-000013-8840 | DFCI-ONCOPANEL-2 |
| GENIE-DFCI-000136-6004 | DFCI-ONCOPANEL-2 |
| GENIE-DFCI-000215-8010 | DFCI-ONCOPANEL-2 |
| GENIE-DFCI-000381-9526 | DFCI-ONCOPANEL-2 |
| GENIE-DFCI-000410-10003 | DFCI-ONCOPANEL-2 |
| GENIE-DFCI-000583-11175 | DFCI-ONCOPANEL-2 |

# Not All Patients Tested on Same Panel

## Without Panel Annotation

```r
1  binmat1 <- gnomeR::create_gene_binary(
2    mutation = mutations,
3    cna = reformat_cna,
4    fusion = reformat_fusions,
5    samples = nsclc_samp$cpt_genie_sample_id,
6    specify_panel = "no")
```

| sample_id | panel_id | GLI2 | KRAS | PIK3C2B | PBRM1 |
|---|---|---|---|---|---|
| GENIE-DFCI-004022-1313 | DFCI-ONCOPANEL-1 | 0 | 0 | 0 | 0 |
| GENIE-DFCI-000013-8840 | DFCI-ONCOPANEL-2 | 0 | 1 | 0 | 0 |
| GENIE-MSK-P-0002725-T01-IM3 | MSK-IMPACT341 | 0 | 1 | 0 | 0 |
| GENIE-MSK-P-0017722-T02-IM6 | MSK-IMPACT468 | 0 | 0 | 0 | 0 |

## With Panel Annotation

```r
1  binmat2 <- gnomeR::create_gene_binary(
2    mutation = mutations,
3    cna = reformat_cna,
4    fusion = reformat_fusions,
5    samples = nsclc_samp$cpt_genie_sample_id,
6    specify_panel = nsclc_panels)
```

| sample_id | panel_id | GLI2 | KRAS | PIK3C2B | PBRM1 |
|---|---|---|---|---|---|
| GENIE-DFCI-004022-1313 | DFCI-ONCOPANEL-1 | 0 | 0 | 0 | NA |
| GENIE-DFCI-000013-8840 | DFCI-ONCOPANEL-2 | 0 | 1 | 0 | 0 |
| GENIE-MSK-P-0002725-T01-IM3 | MSK-IMPACT341 | NA | 1 | NA | 0 |
| GENIE-MSK-P-0017722-T02-IM6 | MSK-IMPACT468 | NA | 0 | NA | 0 |

# Ensure Gene Names Are Consistent Across Studies

> ℹ 2. **Data Formats & Gene Standards Often Inconsistent**
>
> - Column names, data formats and gene names may differ between studies or even within studies.

```
1  no_recode <- gnomeR::create_gene_binary(
2    samples = nsclc_samp$cpt_genie_sample_id,
3    mutation = mutations,
4    cna = reformat_cna,
5    fusion = reformat_fusions,
6    specify_panel = nsclc_panels,
7    recode_aliases = "no")
```

```
1  recode <- gnomeR::create_gene_binary(
2    samples = nsclc_samp$cpt_genie_sample_id,
3    mutation = mutations,
4    cna = reformat_cna,
5    fusion = reformat_fusions,
6    specify_panel = nsclc_panels,
7    recode_aliases = "impact")
```

```
1  setdiff(names(no_recode), names(recode)) %>% head()
```

[1] "MRE11A" "RFWD2" "H3F3A" "FAM46C" "HIST1H3D" "WHSC1L1"

# Process Data: Final Dataset

Let's run `create_gene_binary()` with the `samples`, `specify_panel` and `recode_aliases` arguments.

First we create `nsclc_panels`:

```
1  nsclc_panels <- data.frame(
2    sample_id = nsclc_samp$cpt_genie_sample_id,
3    panel_id = nsclc_samp$cpt_seq_assay_id) %>%
4    mutate(panel_id = ifelse(!is.na(panel_id),
5                             panel_id, "no"))
```

Then run `create_gene_binary()`:

```
1  gene_binary <- gnomeR::create_gene_binary(
2    mutation = mutations,
3    cna = reformat_cna,
4    fusion = reformat_fusions,
5    samples = nsclc_samp$cpt_genie_sample_id,
6    specify_panel = nsclc_panels,
7    recode_aliases = "impact")
```

# Analyzing Data

# Issues When Analyzing Multi-Institutional Genomic Data

# Issues When Analyzing Multi-Institutional Genomic Data

1. **Multiple Testing and False Positives**

# Issues When Analyzing Multi-Institutional Genomic Data

1. **Multiple Testing and False Positives**

- Many hypothesis tests done simultaneously can lead to false positive findings.

# Issues When Analyzing Multi-Institutional Genomic Data

1. **Multiple Testing and False Positives**

- Many hypothesis tests done simultaneously can lead to false positive findings.
- Very low prevalence genes are often not very informative.

# Issues When Analyzing Multi-Institutional Genomic Data

1. **Multiple Testing and False Positives**

   - Many hypothesis tests done simultaneously can lead to false positive findings.

   - Very low prevalence genes are often not very informative.

   - Choose a threshold (e.g. 1% or 5 %) *a priori* and consider reporting a q-value (adjusted for multiple testing).

# Issues When Analyzing Multi-Institutional Genomic Data

1. **Multiple Testing and False Positives**

- Many hypothesis tests done simultaneously can lead to false positive findings.

- Very low prevalence genes are often not very informative.

- Choose a threshold (e.g. 1% or 5 %) *a priori* and consider reporting a q-value (adjusted for multiple testing).

2. **Limited Power To Detect Clinical Associations When Sparse Alterations**

# Issues When Analyzing Multi-Institutional Genomic Data

## 1. Multiple Testing and False Positives

- Many hypothesis tests done simultaneously can lead to false positive findings.

- Very low prevalence genes are often not very informative.

- Choose a threshold (e.g. 1% or 5 %) *a priori* and consider reporting a q-value (adjusted for multiple testing).

## 2. Limited Power To Detect Clinical Associations When Sparse Alterations

- If biologically meaningful, you may want to summarize on gene or pathway level

# Case Study

# Case Study

- We will use the processed binary data frame data (`gene_binary`) to **summarize genomic alterations** overall in the cohort, and **by sex**.

# Case Study

- We will use the processed binary data frame data (`gene_binary`) to **summarize genomic alterations** overall in the cohort, and **by sex**.

- First, we need to join clinical data on `sex` to genomic data

# Add Clinical Variable To Data

```r
1   # get patient IDs and sample IDs
2   patient_index <-  nsclc_cohort$cohort_ngs %>%
3     select(record_id, cpt_genie_sample_id)
4
5   # Join sex data to patient ID index
6   select_clinical <- nsclc_cohort$cohort_pt_char %>%
7     select(record_id, naaccr_sex_code) %>%
8     left_join(patient_index)
9
10  # Join all to gene binary data
11  gene_binary <- gene_binary %>%
12    left_join(select_clinical,
13              by = c("sample_id"= "cpt_genie_sample_id")) %>%
14    select(-record_id)
15
16  gene_binary <- gene_binary %>%
17    select(sample_id, naaccr_sex_code, everything())
18
19  gene_binary %>%
20    select(naaccr_sex_code) %>%
21    tbl_summary()
```

| Characteristic | N = 241[1] |
|---|---|
| naaccr_sex_code | |
| Female | 145 (60%) |
| Male | 96 (40%) |
| [1] n (%) | |

# Subset By a Prevalence Threshold

> (i) 1. **Multiple Testing and False Positives**

- Use `subset_by_frequency(t)` to **subset genes** above a given threshold
- `t` indicates a **prevalence threshold** between 0 (`t = 0`) and 100% (`t = 1`)
- `other_vars` retains the clinical variable of interest in the resulting data set

# Subset By 40% Threshold

```r
ncol(gene_binary)
```

[1] 1403

```r
nsclc_subset <- gene_binary %>%
  subset_by_frequency(t = .4, other_vars = naaccr_sex_code)

ncol(nsclc_subset)
```

[1] 6

# Subset by a Panel

Use `subset_by_panel()` to subset genes in a given targeted panel.

```
1  ncol(gene_binary)
```

[1] 1403

```
1  nsclc_subset_panel <- gene_binary %>%
2    subset_by_panel(panel_id = 'IMPACT300', other_vars = naaccr_sex_code)
3
4  ncol(nsclc_subset_panel)
```

[1] 220

# Summarize Alterations with tbl_genomic()

- `tbl_genomic()` is a wrapper function for `gtsummary::tbl_summary()` specifically designed for presenting genomic data

- You can use any **{gtsummary}** function on top of `tbl_genomic()` to customize the table (e.g. `bold_labels()`)

# Summarize Alterations with tbl_genomic

Create a simple `tbl_genomic` object, then bold the labels.

```
1  nsclc_subset %>%
2    select(-naaccr_sex_code) %>%
3    tbl_genomic() %>%
4    bold_labels()
```

| Characteristic | N = 241[1] |
|---|---|
| **TP53** | 124 (51%) |
| **GBA.Amp** | 17 (46%) |
| Unknown | 204 |
| **KRAS** | 98 (41%) |
| **JAZF1.Amp** | 15 (41%) |
| Unknown | 204 |
| [1] n (%) | |

# Summarize Genes with tbl_genomic()

You may want to analyze on the **gene level** instead of the alteration level.

Use `summarize_by_gene()` first, then pass to `tbl_genomic()`:

> ⚠️ **Note:** `summarize_by_gene()` should come before passing to `subset_by_frequency()`

```r
tbl_gene <- gene_binary %>%
  select(-naaccr_sex_code) %>%
  summarize_by_gene() %>%
  subset_by_frequency(t = .4) %>%
  tbl_genomic()

tbl_gene
```
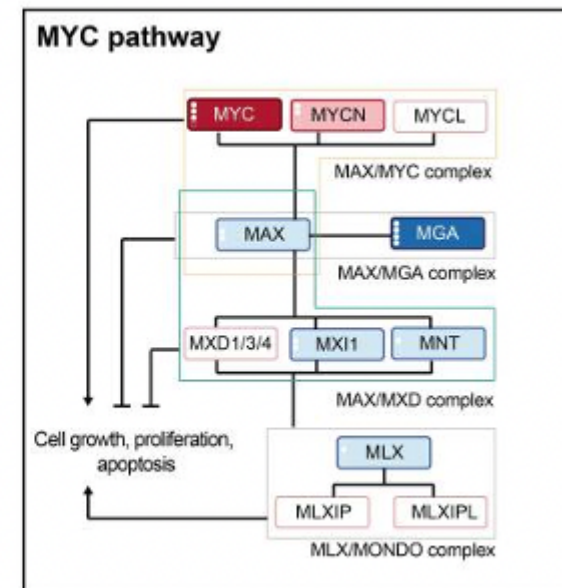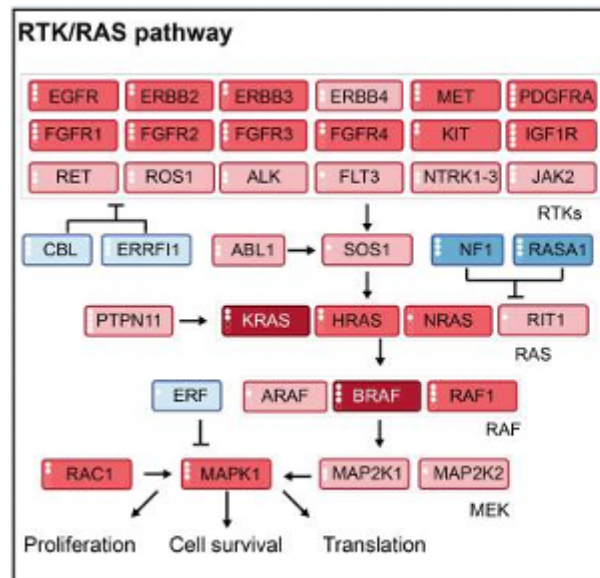
| Characteristic | N = 241[1] |
|---|---|
| TP53 | 135 (56%) |
| GBA | 17 (46%) |
| Unknown | 204 |
| PTK2B | 17 (46%) |
| Unknown | 204 |
| KRAS | 107 (44%) |
| WRN | 38 (41%) |
| Unknown | 149 |
| JAZF1 | 15 (41%) |
| Unknown | 204 |
| [1] n (%) | |

# Summarize Oncogenic Pathways

> (i) 2. **Limited Power To Detect Clinical Associations When Sparse Alterations**
>
> - If biologically meaningful, you may want to summarize on Alteration, Gene or Pathway Level

Often we want to analyze alterations on the pathway level.



Sanchez-Vega F, et al. Oncogenic Signaling Pathways in The Cancer Genome Atlas. Cell. 2018 Apr 5;173(2):321-337.e10. doi:

# Summarize Oncogenic Pathways

> ⓘ 2. **Limited Power To Detect Clinical Associations When Sparse Alterations**
>
> - If biologically meaningful, you may want to summarize on Alteration, Gene or Pathway Level

{gnomeR} offers several default pathways that can be added with `add_pathways()`

```r
1  paths <- gnomeR::pathways %>% names()
2  paths
```

[1] "RTK/RAS" "Nrf2" "PI3K" "TGFB" "p53"
[6] "Wnt" "Myc" "Cell cycle" "Hippo" "Notch"

# Summarize Oncogenic Pathways

You can also add a custom pathway:

```r
1  path_df <- gene_binary %>%
2     add_pathways(custom_pathways = c("SPOP.mut", "FOXA1.mut"))
3
4  path_df %>% select("pathway_custom") %>%
5     tbl_summary()
```

| Characteristic | N = 241[1] |
|---|---|
| pathway_custom | 2 (0.8%) |

[1] n (%)

Note: You must specify `.mut`, `.Amp`, `.Del` for alterations `custom_pathways`

# Summarize Alteration Pathways

```r
path_df <- gene_binary %>%
  select(-naaccr_sex_code) %>%
  add_pathways()

path_df %>%
  select(contains("pathway")) %>%
  tbl_summary() %>%
  bold_labels()
```

# Summarize Alteration Pathways

```r
path_df <- gene_binary %>%
  select(-naaccr_sex_code) %>%
  add_pathways()

path_df %>%
  select(contains("pathway")) %>%
  tbl_summary() %>%
  bold_labels()
```

| Characteristic | N = 241[1] |
|---|---|
| pathway_RTK/RAS | 217 (90%) |
| pathway_Nrf2 | 48 (20%) |
| pathway_PI3K | 114 (47%) |
| pathway_TGFB | 34 (14%) |
| pathway_p53 | 181 (75%) |
| pathway_Wnt | 51 (21%) |
| pathway_Myc | 70 (29%) |
| pathway_Cell cycle | 90 (37%) |
| pathway_Hippo | 31 (13%) |
| pathway_Notch | 72 (30%) |
| [1] n (%) | |

# Comparing Alteration Frequencies Across Clinical Data

We can easily compare frequencies by sex using the by argument:

```r
1  tbl_gene <- gene_binary %>%
2    subset_by_frequency(
3      t = .4,
4      other_vars = naaccr_sex_code) %>%
5    tbl_genomic(by = naaccr_sex_code) %>%
6    bold_labels()
7
8  tbl_gene
```

# Comparing Alteration Frequencies Across Clinical Data

We can easily compare frequencies by sex using the by argument:

```r
1  tbl_gene <- gene_binary %>%
2    subset_by_frequency(
3      t = .4,
4      other_vars = naaccr_sex_code) %>%
5    tbl_genomic(by = naaccr_sex_code) %>%
6    bold_labels()
7
8  tbl_gene
```

| Characteristic | Overall, N = 241[1] | Female, N = 145[1] | Male, N = 96[1] |
|---|---|---|---|
| **TP53** | 124 (51%) | 72 (50%) | 52 (54%) |
| **GBA.Amp** | 17 (46%) | 8 (36%) | 9 (60%) |
| Unknown | 204 | 123 | 81 |
| **KRAS** | 98 (41%) | 66 (46%) | 32 (33%) |
| **JAZF1.Amp** | 15 (41%) | 9 (41%) | 6 (40%) |
| Unknown | 204 | 123 | 81 |

[1] n (%)

# Comparing Alteration Frequencies Across Clinical Data

> ⓘ  1. **Multiple Testing and False Positives**

We can use {gtsummary}'s add_p( ) and add_q( ) for hypothesis testing

```
1   tbl_gene <- gene_binary %>%
2     subset_by_frequency(
3       t = .4,
4       other_vars = naaccr_sex_code) %>%
5     tbl_genomic(
6       by = naaccr_sex_code) %>%
7     bold_labels() %>%
8     add_p() %>%
9     add_q()
10
11  tbl_gene
```

| Characteristic | Overall, N = 241[1] | Female, N = 145[1] | Male, N = 96[1] | p-value[2] | q-value[3] |
|---|---|---|---|---|---|
| **TP53** | 124 (51%) | 72 (50%) | 52 (54%) | 0.5 | 0.7 |
| **GBA.Amp** | 17 (46%) | 8 (36%) | 9 (60%) | 0.2 | 0.3 |
| Unknown | 204 | 123 | 81 | | |
| **KRAS** | 98 (41%) | 66 (46%) | 32 (33%) | 0.059 | 0.2 |
| **JAZF1.Amp** | 15 (41%) | 9 (41%) | 6 (40%) | >0.9 | >0.9 |
| Unknown | 204 | 123 | 81 | | |

[1] n (%)

[2] Pearson's Chi-squared test

[3] False discovery rate correction for multiple testing

# Comparing Alteration Frequencies Across Clinical Data

```r
 1  tbl_gene <- gene_binary %>%
 2    subset_by_frequency(
 3      t = .4,
 4      other_vars = naaccr_sex_code) %>%
 5    tbl_genomic(by = naaccr_sex_code) %>%
 6    bold_labels() %>%
 7    add_p() %>%
 8    add_q()
 9
10  tbl_gene
```

| Characteristic | Overall, N = 241[1] | Female, N = 145[1] | Male, N = 96[1] | p-value[2] | q-value[3] |
|---|---|---|---|---|---|
| **TP53** | 124 (51%) | 72 (50%) | 52 (54%) | 0.5 | 0.7 |
| **GBA.Amp** | 17 (46%) | 8 (36%) | 9 (60%) | 0.2 | 0.3 |
| Unknown | 204 | 123 | 81 | | |
| **KRAS** | 98 (41%) | 66 (46%) | 32 (33%) | 0.059 | 0.2 |
| **JAZF1.Amp** | 15 (41%) | 9 (41%) | 6 (40%) | >0.9 | >0.9 |
| Unknown | 204 | 123 | 81 | | |

[1] n (%)

[2] Pearson's Chi-squared test

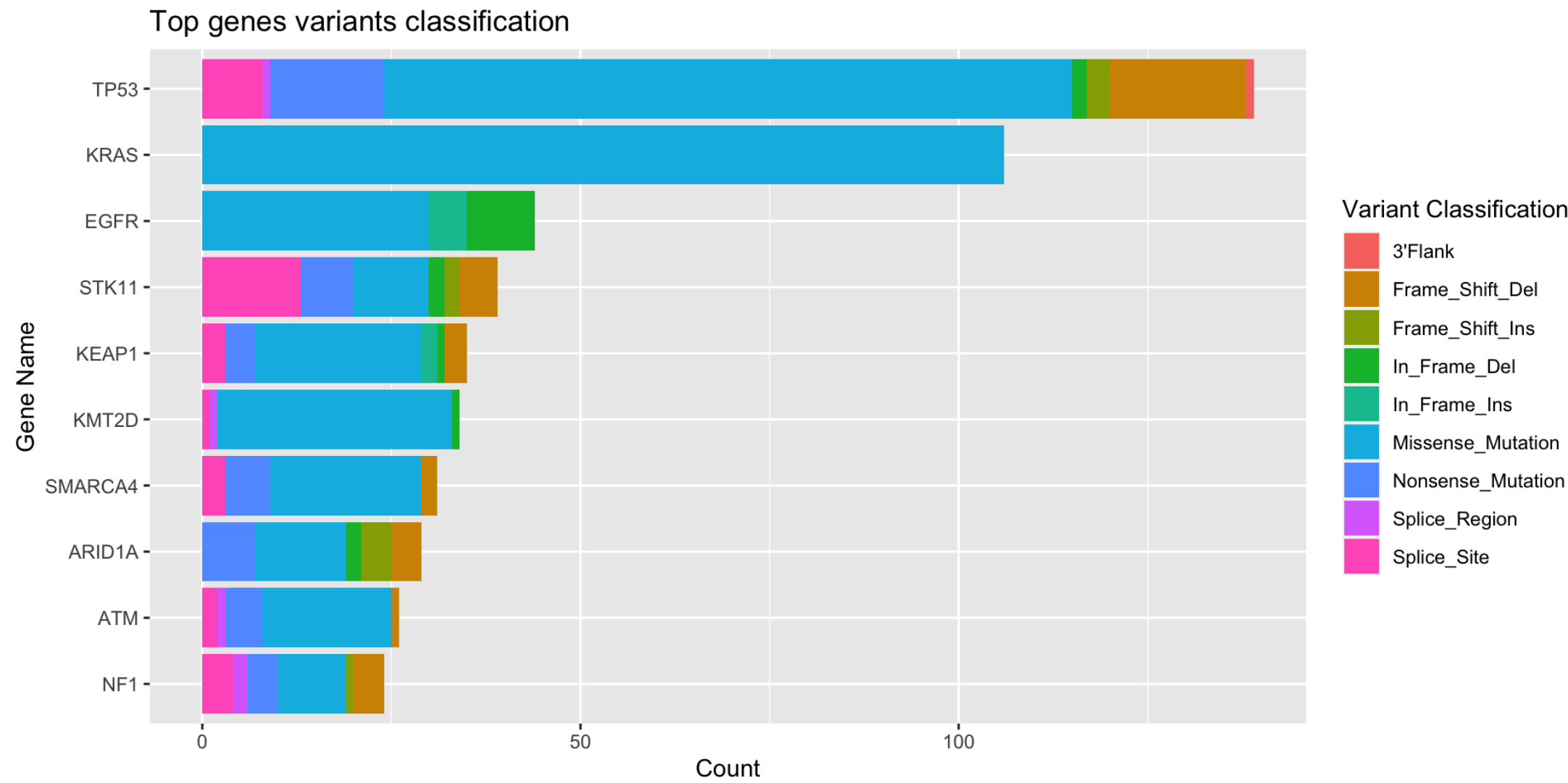[3] False discovery rate correction for multiple testing

# Visualize Data

```
1  gnomeR::ggcomut(nsclc_cohort$cohort_mutations_extended)
```
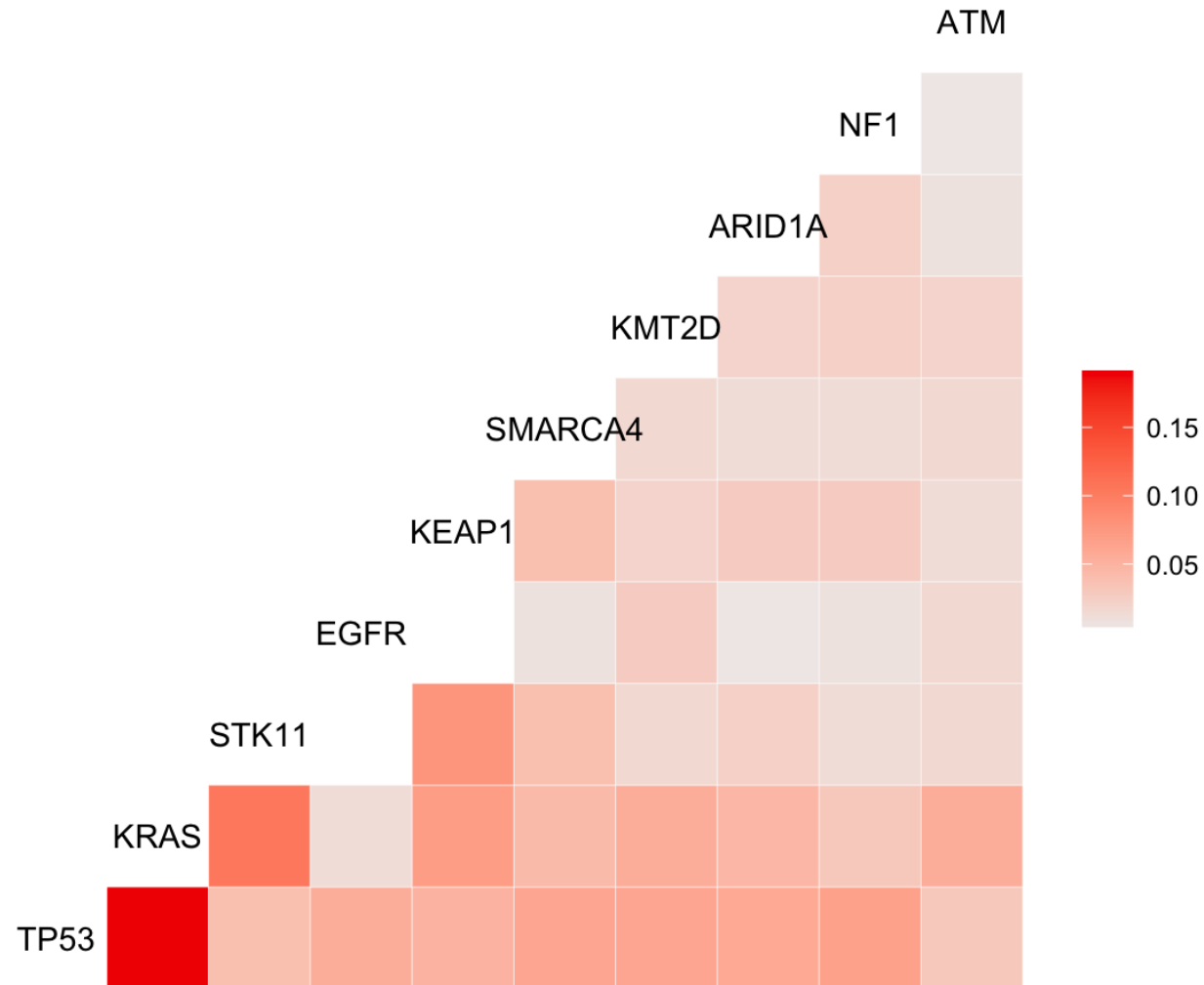
# Visualize Data

```
1  gnomeR::ggtopgenes(nsclc_cohort$cohort_mutations_extended)
```



Top genes variants classification

# Additional Items & Next Steps

- Additional **visualizations** and **color palettes** useful for genomic data are available in {gnomeR} package

- Some data may require additional data checks. See {gnomeR vignette} for helpful tips on data QA.

- It may be appropriate to **oncoKB annotate** your data and only analyze oncogenic mutations (see oncoKB.org for more information).

# Conclusion

- The **{genieBPC}** & **{gnomeR}** R packages offer a reproducible pipeline to create cohorts for clinico-genomics analyses

- **{genieBPC}** streamlines data access and clinical data processing from multiple clinical data files of varying structure to create analytic cohorts

- **{gnomeR}** facilitates annotation and analysis of complicated genomic data.



*Note: {gnomeR} can be used for genomic data processing and analyses outside of the GENIE BPC project!*

# Thank You!

Thanks to all {gnomeR} & {genieBPC} authors & contributors:

Michael Curry, Axel Martin, Arshi Arora, Allie Reiner, Akriti Mishra, Dan Sjoberg, Jessica Flynn, Caroline Kostrzewa, Christine Zhou, Esther Drill, Margie Hannum, and more.