csa-atrophy

Release v1.0

Paul Bautin

Jun 01, 2021

HOW TO RUN

1 How to run? 3

2 Introduction 5

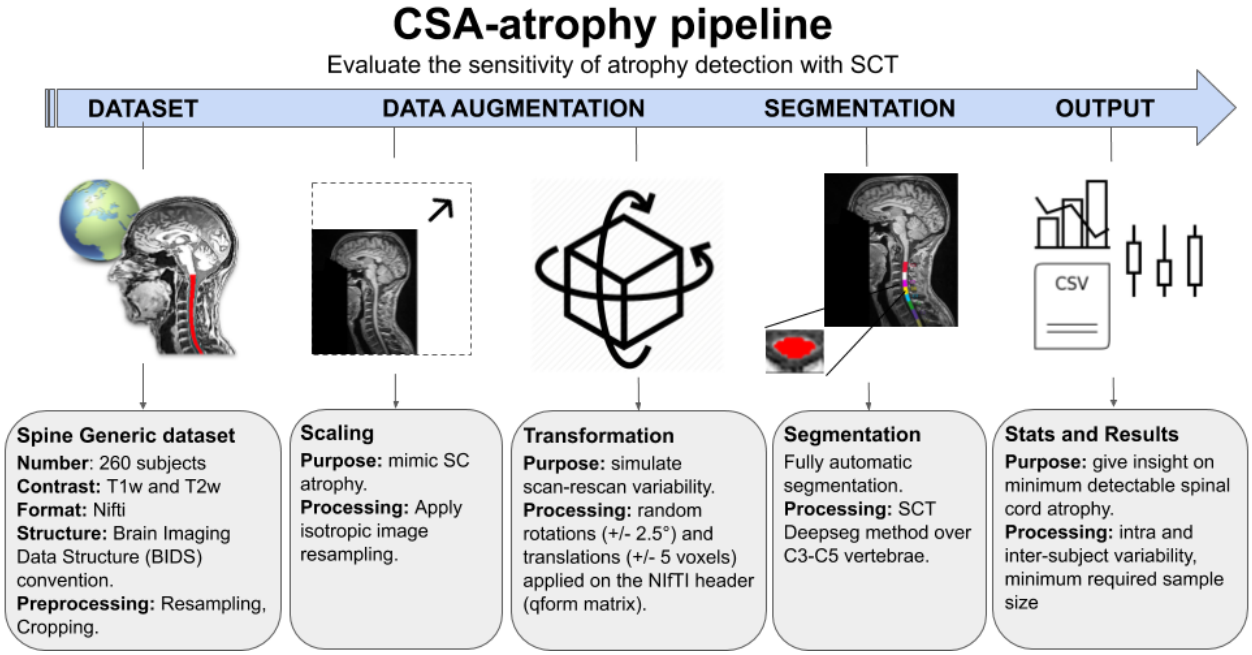
3 Intra-subject 7

4 Inter-subject 11

5 Power analysis 17

6 Quality control 19

CSA-atrophy evaluates the robustness and the sensitivity of an automated analysis pipeline for detecting SC atrophy. Notably, the proposed framework utilizes image scaling and applies a random rigid transformation to mimic subject repositioning (scan-rescan). This enables the quantification of the accuracy and precision of the estimated CSA across various degrees of simulated atrophy. As presented in section statistics, statistics from these experiments such as power analyses and minimum sample sizes are derived.



CHAPTER

ONE

HOW TO RUN?

This code has been tested using Python 3.7.

Download (or git clone) this repository:

```
git clone https://github.com/sct-pipeline/csa-atrophy.git
cd csa-atrophy
```

Installation: csa-atrophy requires specific python packages for computing statistics and processing images. If not already present on the computer's python environment such packages will automatically be installed by running pip command:

```
pip install -e .
```

Download the results file from Spine Generic Multi-Subject dataset: https://github.com/spine-generic/data-multi-subject/releases/tag/r20201130 .

Edit the file *config_sct_run_batch.yml* according to your setup. Notable flags include:

- *path_data*: If you downloaded the spine-generic data at another location, make sure to update the path;
- *include_list*: If you only want to run the script in a few subjects, list them here. Example: *include_list: ['sub-unf04', 'sub-unf05']*

See *sct_run_batch -h* to look at the available options.

Run the analysis:

```
sct_run_batch -config config_sct_run_batch.yml
```

note: desired subjects using flag -include and in parallel processing using flag -jobs.

To output statistics, run in Dataset

```
csa_rescale_stat -i csa_atrophy_results/results -o csa_atrophy_results -config config_
↪script.yml -fig
```

CHAPTER

TWO

INTRODUCTION

Documentation on statistics to evaluate the sensitivity of atrophy detection with SCT. Average CSA of each image is indexed as $CSA(sI, rX, tY)$ where:

- (sI) corresponds to subject I
- (rX) corresponds to the applied scaling factor X on the native image (e.g. X=1, X=0.95, X=0.8)
- (tY) corresponds to the applied random affine transformation Y on the native image

CHAPTER

THREE

INTRA-SUBJECT

Intra-subject statistics. These statistics are gathered per rescaling and per subject in the Panda dataframe `df_sub`:

3.1 Intra-subject CSA (CSA estimation)

CSA averaged across transformations.

$\mu_t\{CSA_{sI,rX}\}$

```
    print("\n==================== subject_dataframe ========================\n")
    df_sub = pd.DataFrame()
    # add necessary columns to df_sub dataframe
    df_sub['rescale'] = df.groupby(['rescale', 'subject']).mean().reset_index()['rescale
↪']
    df_sub['rescale_area'] = 100 * (df.groupby(['rescale', 'subject']).mean().reset_
↪index()['rescale'] ** 2)
    df_sub['subject'] = df.groupby(['rescale', 'subject']).mean().reset_index()['subject
↪']
    df_sub['num_tf'] = df.groupby(['rescale', 'subject'])['transfo'].count().values
    df_sub['num_slices'] = df.groupby(['rescale', 'subject'])['num_slices'].mean().values
    # add stats to per subject dataframe
    df_sub['mean'] = df.groupby(['rescale', 'subject']).mean()['MEAN(area)'].values
    df_sub['std'] = df.groupby(['rescale', 'subject']).std()['MEAN(area)'].values
    df_sub['cov'] = df_sub['std'].div(df_sub['mean'])
    df_sub = add_columns_df_sub(df_sub)
    df_sub['rescale_estimated'] = df_sub['mean'].div(df_sub['csa_without_rescale'])
    df_sub['error'] = (df_sub['mean'] - df_sub['theoretic_csa']).abs()
    df_sub['perc_error'] = 100 * (df_sub['mean'] - df_sub['theoretic_csa']).abs().div(df_
↪sub['theoretic_csa'])
    print(df_sub)
    # save dataframe in a csv file
    df_sub.to_csv(os.path.join(path_output, r'csa_sub.csv'))
```

3.2 Intra-subject SD

SD of CSA across transformations.

$\sigma_t\{CSA_{sI,rX}\}$

```
    print("\n==================== subject_dataframe ==========================\n")
    df_sub = pd.DataFrame()
    # add necessary columns to df_sub dataframe
    df_sub['rescale'] = df.groupby(['rescale', 'subject']).mean().reset_index()['rescale
↪']
    df_sub['rescale_area'] = 100 * (df.groupby(['rescale', 'subject']).mean().reset_
↪index()['rescale'] ** 2)
    df_sub['subject'] = df.groupby(['rescale', 'subject']).mean().reset_index()['subject
↪']
    df_sub['num_tf'] = df.groupby(['rescale', 'subject'])['transfo'].count().values
    df_sub['num_slices'] = df.groupby(['rescale', 'subject'])['num_slices'].mean().values
    # add stats to per subject dataframe
    df_sub['mean'] = df.groupby(['rescale', 'subject']).mean()['MEAN(area)'].values
    df_sub['std'] = df.groupby(['rescale', 'subject']).std()['MEAN(area)'].values
    df_sub['cov'] = df_sub['std'].div(df_sub['mean'])
    df_sub = add_columns_df_sub(df_sub)
    df_sub['rescale_estimated'] = df_sub['mean'].div(df_sub['csa_without_rescale'])
    df_sub['error'] = (df_sub['mean'] - df_sub['theoretic_csa']).abs()
    df_sub['perc_error'] = 100 * (df_sub['mean'] - df_sub['theoretic_csa']).abs().div(df_
↪sub['theoretic_csa'])
    print(df_sub)
    # save dataframe in a csv file
    df_sub.to_csv(os.path.join(path_output, r'csa_sub.csv'))
```

3.3 Intra-subject COV

COV of CSA across transformations.

$COV_t\{CSA_{sI,rX}\}$

```
    print("\n==================== subject_dataframe ==========================\n")
    df_sub = pd.DataFrame()
    # add necessary columns to df_sub dataframe
    df_sub['rescale'] = df.groupby(['rescale', 'subject']).mean().reset_index()['rescale
↪']
    df_sub['rescale_area'] = 100 * (df.groupby(['rescale', 'subject']).mean().reset_
↪index()['rescale'] ** 2)
    df_sub['subject'] = df.groupby(['rescale', 'subject']).mean().reset_index()['subject
↪']
    df_sub['num_tf'] = df.groupby(['rescale', 'subject'])['transfo'].count().values
    df_sub['num_slices'] = df.groupby(['rescale', 'subject'])['num_slices'].mean().values
    # add stats to per subject dataframe
    df_sub['mean'] = df.groupby(['rescale', 'subject']).mean()['MEAN(area)'].values
    df_sub['std'] = df.groupby(['rescale', 'subject']).std()['MEAN(area)'].values
    df_sub['cov'] = df_sub['std'].div(df_sub['mean'])
    df_sub = add_columns_df_sub(df_sub)
```

(continues on next page)

(continued from previous page)

```
    df_sub['rescale_estimated'] = df_sub['mean'].div(df_sub['csa_without_rescale'])
    df_sub['error'] = (df_sub['mean'] - df_sub['theoretic_csa']).abs()
    df_sub['perc_error'] = 100 * (df_sub['mean'] - df_sub['theoretic_csa']).abs().div(df_
↪sub['theoretic_csa'])
    print(df_sub)
    # save dataframe in a csv file
    df_sub.to_csv(os.path.join(path_output, r'csa_sub.csv'))
```

3.4 Rescale estimation (RE)

ratio of the atrophied CSA divided by the un-rescaled CSA averaged across transformations (gives an estimation of the applied scaling).

$\mu_t \left\{ \frac{CSA_{sI,rX}}{CSA_{sI,r1}} \right\}$

```
    print("\n==================== subject_dataframe ========================\n")
    df_sub = pd.DataFrame()
    # add necessary columns to df_sub dataframe
    df_sub['rescale'] = df.groupby(['rescale', 'subject']).mean().reset_index()['rescale
↪']
    df_sub['rescale_area'] = 100 * (df.groupby(['rescale', 'subject']).mean().reset_
↪index()['rescale'] ** 2)
    df_sub['subject'] = df.groupby(['rescale', 'subject']).mean().reset_index()['subject
↪']
    df_sub['num_tf'] = df.groupby(['rescale', 'subject'])['transfo'].count().values
    df_sub['num_slices'] = df.groupby(['rescale', 'subject'])['num_slices'].mean().values
    # add stats to per subject dataframe
    df_sub['mean'] = df.groupby(['rescale', 'subject']).mean()['MEAN(area)'].values
    df_sub['std'] = df.groupby(['rescale', 'subject']).std()['MEAN(area)'].values
    df_sub['cov'] = df_sub['std'].div(df_sub['mean'])
    df_sub = add_columns_df_sub(df_sub)
    df_sub['rescale_estimated'] = df_sub['mean'].div(df_sub['csa_without_rescale'])
    df_sub['error'] = (df_sub['mean'] - df_sub['theoretic_csa']).abs()
    df_sub['perc_error'] = 100 * (df_sub['mean'] - df_sub['theoretic_csa']).abs().div(df_
↪sub['theoretic_csa'])
    print(df_sub)
    # save dataframe in a csv file
    df_sub.to_csv(os.path.join(path_output, r'csa_sub.csv'))
```

3.5 Error

mean absolute error on CSA estimation averaged across transformations.

$\mu_t\{CSA_{sI,rX}\} - \mu_t\{CSA_{sI,r1} \cdot (rX)^2\}$

```
    print("\n==================== subject_dataframe ========================\n")
    df_sub = pd.DataFrame()
    # add necessary columns to df_sub dataframe
    df_sub['rescale'] = df.groupby(['rescale', 'subject']).mean().reset_index()['rescale
↪']
```

(continues on next page)

(continued from previous page)

```
    df_sub['rescale_area'] = 100 * (df.groupby(['rescale', 'subject']).mean().reset_
↪index()['rescale'] ** 2)
    df_sub['subject'] = df.groupby(['rescale', 'subject']).mean().reset_index()['subject
↪']
    df_sub['num_tf'] = df.groupby(['rescale', 'subject'])['transfo'].count().values
    df_sub['num_slices'] = df.groupby(['rescale', 'subject'])['num_slices'].mean().values
    # add stats to per subject dataframe
    df_sub['mean'] = df.groupby(['rescale', 'subject']).mean()['MEAN(area)'].values
    df_sub['std'] = df.groupby(['rescale', 'subject']).std()['MEAN(area)'].values
    df_sub['cov'] = df_sub['std'].div(df_sub['mean'])
    df_sub = add_columns_df_sub(df_sub)
    df_sub['rescale_estimated'] = df_sub['mean'].div(df_sub['csa_without_rescale'])
    df_sub['error'] = (df_sub['mean'] - df_sub['theoretic_csa']).abs()
    df_sub['perc_error'] = 100 * (df_sub['mean'] - df_sub['theoretic_csa']).abs().div(df_
↪sub['theoretic_csa'])
    print(df_sub)
    # save dataframe in a csv file
    df_sub.to_csv(os.path.join(path_output, r'csa_sub.csv'))
```

CHAPTER

FOUR

INTER-SUBJECT

Inter-subject statistics. These statistics are gathered per scaling in the Panda dataframe `df_rescale`

4.1 Mean intra-subject SD

Intra-subject SD averaged across subjects.

$\mu_s\{\sigma_t\{CSA_{rX}\}\}$

```
    print("\n==================== rescaling_dataframe ==========================\n")
    df_rescale = pd.DataFrame()
    df_rescale['rescale'] = df_sub.groupby(['rescale']).mean().reset_index()['rescale']
    df_rescale['rescale_area'] = df_sub.groupby('rescale_area').mean().reset_index()[
↪'rescale_area']
    df_rescale['mean_slices'] = df_sub.groupby(['rescale']).mean()['num_slices'].values
    df_rescale['std_slices'] = df_sub.groupby(['rescale']).std()['num_slices'].values
    df_rescale['num_sub'] = df_sub.groupby('rescale')['mean'].count().values
    df_rescale['mean_inter'] = df_sub.groupby('rescale').mean()['mean'].values
    df_rescale['std_intra'] = df_sub.groupby('rescale').mean()['std'].values
    df_rescale['cov_intra'] = df_sub.groupby('rescale').mean()['cov'].values
    df_rescale['std_inter'] = df_sub.groupby('rescale').std()['mean'].values
    df_rescale['mean_rescale_estimated'] = df_sub.groupby('rescale').mean()['rescale_
↪estimated'].values
    df_rescale['std_rescale_estimated'] = df_sub.groupby('rescale').std()['rescale_
↪estimated'].values
    df_rescale['mean_perc_error'] = df_sub.groupby('rescale').mean()['perc_error'].values
    df_rescale['mean_error'] = df_sub.groupby('rescale').mean()['error'].values
    df_rescale['std_perc_error'] = df_sub.groupby('rescale').std()['perc_error'].values
    df_rescale['sample_size'] = sample_size(df_rescale, config_param)
    print(df_rescale)
```

4.2 Mean intra-subject COV

Intra-subject COV averaged across subjects.

$\mu_s\{COV_t\{CSA_{rX}\}\}$

```
    print("\n==================== rescaling_dataframe ==========================\n")
    df_rescale = pd.DataFrame()
    df_rescale['rescale'] = df_sub.groupby(['rescale']).mean().reset_index()['rescale']
    df_rescale['rescale_area'] = df_sub.groupby('rescale_area').mean().reset_index()[
↪'rescale_area']
    df_rescale['mean_slices'] = df_sub.groupby(['rescale']).mean()['num_slices'].values
    df_rescale['std_slices'] = df_sub.groupby(['rescale']).std()['num_slices'].values
    df_rescale['num_sub'] = df_sub.groupby('rescale')['mean'].count().values
    df_rescale['mean_inter'] = df_sub.groupby('rescale').mean()['mean'].values
    df_rescale['std_intra'] = df_sub.groupby('rescale').mean()['std'].values
    df_rescale['cov_intra'] = df_sub.groupby('rescale').mean()['cov'].values
    df_rescale['std_inter'] = df_sub.groupby('rescale').std()['mean'].values
    df_rescale['mean_rescale_estimated'] = df_sub.groupby('rescale').mean()['rescale_
↪estimated'].values
    df_rescale['std_rescale_estimated'] = df_sub.groupby('rescale').std()['rescale_
↪estimated'].values
    df_rescale['mean_perc_error'] = df_sub.groupby('rescale').mean()['perc_error'].values
    df_rescale['mean_error'] = df_sub.groupby('rescale').mean()['error'].values
    df_rescale['std_perc_error'] = df_sub.groupby('rescale').std()['perc_error'].values
    df_rescale['sample_size'] = sample_size(df_rescale, config_param)
    print(df_rescale)
```

4.3 Inter-subject SD

SD of intra-subject CSA across subjects.

$\sigma_s\{\mu_t\{CSA_{rX}\}\}$

```
    print("\n==================== rescaling_dataframe ==========================\n")
    df_rescale = pd.DataFrame()
    df_rescale['rescale'] = df_sub.groupby(['rescale']).mean().reset_index()['rescale']
    df_rescale['rescale_area'] = df_sub.groupby('rescale_area').mean().reset_index()[
↪'rescale_area']
    df_rescale['mean_slices'] = df_sub.groupby(['rescale']).mean()['num_slices'].values
    df_rescale['std_slices'] = df_sub.groupby(['rescale']).std()['num_slices'].values
    df_rescale['num_sub'] = df_sub.groupby('rescale')['mean'].count().values
    df_rescale['mean_inter'] = df_sub.groupby('rescale').mean()['mean'].values
    df_rescale['std_intra'] = df_sub.groupby('rescale').mean()['std'].values
    df_rescale['cov_intra'] = df_sub.groupby('rescale').mean()['cov'].values
    df_rescale['std_inter'] = df_sub.groupby('rescale').std()['mean'].values
    df_rescale['mean_rescale_estimated'] = df_sub.groupby('rescale').mean()['rescale_
↪estimated'].values
    df_rescale['std_rescale_estimated'] = df_sub.groupby('rescale').std()['rescale_
↪estimated'].values
    df_rescale['mean_perc_error'] = df_sub.groupby('rescale').mean()['perc_error'].values
    df_rescale['mean_error'] = df_sub.groupby('rescale').mean()['error'].values
```

(continues on next page)

(continued from previous page)

```
    df_rescale['std_perc_error'] = df_sub.groupby('rescale').std()['perc_error'].values
    df_rescale['sample_size'] = sample_size(df_rescale, config_param)
    print(df_rescale)
```

4.4 Mean rescale estimated (RE)

rescale_estimated averaged across subjects.

$\mu_s \left\{ \mu_t \left\{ \frac{CSA_{rX}}{CSA_{r1}} \right\} \right\}$

```
    print("\n==================== rescaling_dataframe ==========================\n")
    df_rescale = pd.DataFrame()
    df_rescale['rescale'] = df_sub.groupby(['rescale']).mean().reset_index()['rescale']
    df_rescale['rescale_area'] = df_sub.groupby('rescale_area').mean().reset_index()[
↪'rescale_area']
    df_rescale['mean_slices'] = df_sub.groupby(['rescale']).mean()['num_slices'].values
    df_rescale['std_slices'] = df_sub.groupby(['rescale']).std()['num_slices'].values
    df_rescale['num_sub'] = df_sub.groupby('rescale')['mean'].count().values
    df_rescale['mean_inter'] = df_sub.groupby('rescale').mean()['mean'].values
    df_rescale['std_intra'] = df_sub.groupby('rescale').mean()['std'].values
    df_rescale['cov_intra'] = df_sub.groupby('rescale').mean()['cov'].values
    df_rescale['std_inter'] = df_sub.groupby('rescale').std()['mean'].values
    df_rescale['mean_rescale_estimated'] = df_sub.groupby('rescale').mean()['rescale_
↪estimated'].values
    df_rescale['std_rescale_estimated'] = df_sub.groupby('rescale').std()['rescale_
↪estimated'].values
    df_rescale['mean_perc_error'] = df_sub.groupby('rescale').mean()['perc_error'].values
    df_rescale['mean_error'] = df_sub.groupby('rescale').mean()['error'].values
    df_rescale['std_perc_error'] = df_sub.groupby('rescale').std()['perc_error'].values
    df_rescale['sample_size'] = sample_size(df_rescale, config_param)
    print(df_rescale)
```

4.5 SD of rescale estimated

SD of rescale_estimated across subjects.

$\sigma_s \left\{ \mu_t \left\{ \frac{CSA_{rX}}{CSA_{r1}} \right\} \right\}$

```
    print("\n==================== rescaling_dataframe ==========================\n")
    df_rescale = pd.DataFrame()
    df_rescale['rescale'] = df_sub.groupby(['rescale']).mean().reset_index()['rescale']
    df_rescale['rescale_area'] = df_sub.groupby('rescale_area').mean().reset_index()[
↪'rescale_area']
    df_rescale['mean_slices'] = df_sub.groupby(['rescale']).mean()['num_slices'].values
    df_rescale['std_slices'] = df_sub.groupby(['rescale']).std()['num_slices'].values
    df_rescale['num_sub'] = df_sub.groupby('rescale')['mean'].count().values
    df_rescale['mean_inter'] = df_sub.groupby('rescale').mean()['mean'].values
    df_rescale['std_intra'] = df_sub.groupby('rescale').mean()['std'].values
```

(continues on next page)

(continued from previous page)

```
    df_rescale['cov_intra'] = df_sub.groupby('rescale').mean()['cov'].values
    df_rescale['std_inter'] = df_sub.groupby('rescale').std()['mean'].values
    df_rescale['mean_rescale_estimated'] = df_sub.groupby('rescale').mean()['rescale_
↪estimated'].values
    df_rescale['std_rescale_estimated'] = df_sub.groupby('rescale').std()['rescale_
↪estimated'].values
    df_rescale['mean_perc_error'] = df_sub.groupby('rescale').mean()['perc_error'].values
    df_rescale['mean_error'] = df_sub.groupby('rescale').mean()['error'].values
    df_rescale['std_perc_error'] = df_sub.groupby('rescale').std()['perc_error'].values
    df_rescale['sample_size'] = sample_size(df_rescale, config_param)
    print(df_rescale)
```

4.6 Mean error

error on the intra-subject CSA estimation averaged across subjects.

$\mu_s\{\mu_t\{CSA_{rX}\} - \mu_t\{CSA_{r1} \cdot (rX)^2\}\}$

```
    print("\n==================== rescaling_dataframe ==========================\n")
    df_rescale = pd.DataFrame()
    df_rescale['rescale'] = df_sub.groupby(['rescale']).mean().reset_index()['rescale']
    df_rescale['rescale_area'] = df_sub.groupby('rescale_area').mean().reset_index()[
↪'rescale_area']
    df_rescale['mean_slices'] = df_sub.groupby(['rescale']).mean()['num_slices'].values
    df_rescale['std_slices'] = df_sub.groupby(['rescale']).std()['num_slices'].values
    df_rescale['num_sub'] = df_sub.groupby('rescale')['mean'].count().values
    df_rescale['mean_inter'] = df_sub.groupby('rescale').mean()['mean'].values
    df_rescale['std_intra'] = df_sub.groupby('rescale').mean()['std'].values
    df_rescale['cov_intra'] = df_sub.groupby('rescale').mean()['cov'].values
    df_rescale['std_inter'] = df_sub.groupby('rescale').std()['mean'].values
    df_rescale['mean_rescale_estimated'] = df_sub.groupby('rescale').mean()['rescale_
↪estimated'].values
    df_rescale['std_rescale_estimated'] = df_sub.groupby('rescale').std()['rescale_
↪estimated'].values
    df_rescale['mean_perc_error'] = df_sub.groupby('rescale').mean()['perc_error'].values
    df_rescale['mean_error'] = df_sub.groupby('rescale').mean()['error'].values
    df_rescale['std_perc_error'] = df_sub.groupby('rescale').std()['perc_error'].values
    df_rescale['sample_size'] = sample_size(df_rescale, config_param)
    print(df_rescale)
```

4.7 SD of error

SD of error on intra-subject CSA estimation across subjects.

$\sigma_s\{\mu_t\{CSA_{rX}\} - \mu_t\{CSA_{r1} \cdot (rX)^2\}\}$

```
    print("\n==================== rescaling_dataframe ==========================\n")
    df_rescale = pd.DataFrame()
    df_rescale['rescale'] = df_sub.groupby(['rescale']).mean().reset_index()['rescale']
```

(continues on next page)

(continued from previous page)

```
    df_rescale['rescale_area'] = df_sub.groupby('rescale_area').mean().reset_index()[
↪'rescale_area']
    df_rescale['mean_slices'] = df_sub.groupby(['rescale']).mean()['num_slices'].values
    df_rescale['std_slices'] = df_sub.groupby(['rescale']).std()['num_slices'].values
    df_rescale['num_sub'] = df_sub.groupby('rescale')['mean'].count().values
    df_rescale['mean_inter'] = df_sub.groupby('rescale').mean()['mean'].values
    df_rescale['std_intra'] = df_sub.groupby('rescale').mean()['std'].values
    df_rescale['cov_intra'] = df_sub.groupby('rescale').mean()['cov'].values
    df_rescale['std_inter'] = df_sub.groupby('rescale').std()['mean'].values
    df_rescale['mean_rescale_estimated'] = df_sub.groupby('rescale').mean()['rescale_
↪estimated'].values
    df_rescale['std_rescale_estimated'] = df_sub.groupby('rescale').std()['rescale_
↪estimated'].values
    df_rescale['mean_perc_error'] = df_sub.groupby('rescale').mean()['perc_error'].values
    df_rescale['mean_error'] = df_sub.groupby('rescale').mean()['error'].values
    df_rescale['std_perc_error'] = df_sub.groupby('rescale').std()['perc_error'].values
    df_rescale['sample_size'] = sample_size(df_rescale, config_param)
    print(df_rescale)
```

CHAPTER

FIVE

POWER ANALYSIS

5.1 Between-group minimum sample size

The minimum sample size, number of subjects per group (study arm), necessary to detect an atrophy between groups was computed based on a two-sample (unpaired) bilateral t-test using the following formula (Wang and Ji 2020; Wittes 2002):

$n_{unpaired} = \frac{(z_{/2}+z)^2(\sigma_{(:,r1)}+\sigma_{(:,rX)})^2}{\Delta^2_{sub}}$

Where $n_{unpaired}$ is the minimum sample size required to differentiate between groups with a given power (z corresponds to the power z score, e.g. 80% power gives =0.2 and z corresponds to the significance level z score, e.g. 5% level of significance gives =0.05 and $z_{/2}$ group is the difference of the mean CSA between the groups.

```
    # create a dataframe from the csv files
    df_vert = pd.DataFrame(data)
    pd.set_option('display.max_rows', None)

    # identify rows with missing values
    print("Remove rows with missing values...")
    lines_to_drop = df_vert[df_vert['MEAN(area)'] == 'None'].index
    df_vert['subject'] = list(sub.split('data_processed/')[1].split('/anat')[0] for sub␣
↪in df_vert['Filename'])

    # remove rows with missing values
    df_vert = df_vert.drop(df_vert.index[lines_to_drop])
```

5.2 Within-subject minimum sample size

the minimum sample size necessary to detect an atrophy in a within-subject (repeated-measures) study was computed based on a two-sample bilateral paired t-test using the following formula (Altmann et al. 2009):

$n_{paired} = \frac{(z_{/2}+z)^2(\sigma_{diff})^2}{\Delta^2_{sub}}$

Where σ_{diff} is the standard deviation between longitudinal CSA measures across subjects and Δ_{sub} is the mean of the difference between longitudinal CSA measures.

```
    # create a dataframe from the csv files
    df_vert = pd.DataFrame(data)
```

(continues on next page)

(continued from previous page)

```
    pd.set_option('display.max_rows', None)

    # identify rows with missing values
    print("Remove rows with missing values...")
    lines_to_drop = df_vert[df_vert['MEAN(area)'] == 'None'].index
    df_vert['subject'] = list(sub.split('data_processed/')[1].split('/anat')[0] for sub␣
↪in df_vert['Filename'])

    # remove rows with missing values
    df_vert = df_vert.drop(df_vert.index[lines_to_drop])
```

CHAPTER

SIX

QUALITY CONTROL

After running the analysis, check your Quality Control (QC) report by opening the file qc/index.html. Use the "Search" feature of the QC report to quickly jump to segmentations or labeling results. If you spot issues (wrong labeling), add their filenames in the 'config_correction.yml' file (see https://spine-generic.rtfd.io/en/latest/analysis-pipeline.html for further indications). Then, manually create labels in the cord at the level of inter-vertebral discs C1-C2, C2-C3, ..., C4-C5 with the command:

```
manual_correction -config config_correction.yml -path-in csa_atrophy_results/data_
↪processed -path-out PATH_DATA
```

The bash script outputs all manual labelings to the derivatives directory in the dataset path defined in *path_data*. It is now possible to re-run the whole process. With the command below labeling will use the manual corrections that are present in the derivatives/ folder of the dataset, otherwise labeling will be done automatically.

```
sct_run_batch -config config_sct_run_batch.yml
```