```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        from scipy import stats
        import scanpy as sc
        sc.settings.verbosity = 3
        sc.logging.print_header()
        from matplotlib import rcParams
        import snapatac2 as snap

        snap.__version__
```

```
scanpy==1.9.8 anndata==0.10.9 umap==0.5.5 numpy==1.26.4 scipy==1.14.1 pandas==2.1.1 scikit-learn==1.5.1 statsmodels==0.14.1 igraph==0.11.6 pynndescent=
=0.5.11
```

```
Out[1]: '2.7.0'
```

```
In [2]: motifs = snap.tl.motif_enrichment( motifs=snap.datasets.cis_bp(unique=True), regions= {'0':['chr1:182408480-182408981']}, genome_fasta=snap.genome.mm1
```

```
2024-09-15 08:11:19 - INFO - Fetching 1 sequences ...
2024-09-15 08:11:20 - INFO - Computing enrichment ...
100%|██████████| 1165/1165 [00:31<00:00, 37.14it/s]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
File ~/.conda/envs/ds_notebook/lib/python3.12/site-packages/polars/_utils/construction/series.py:309, in _construct_series_with_fallbacks(constructor,
name, values, dtype, strict)
    308 try:
--> 309     return constructor(name, values, strict)
    310 except TypeError:

TypeError: 'float' object cannot be interpreted as an integer

During handling of the above exception, another exception occurred:

TypeError                                 Traceback (most recent call last)
Cell In[2], line 1
----> 1 motifs = snap.tl.motif_enrichment( motifs=snap.datasets.cis_bp(unique=True), regions= {'0':['chr1:182408480-182408981']}, genome_fasta=snap.gen
ome.mm10 )

File ~/.conda/envs/ds_notebook/lib/python3.12/site-packages/snapatac2/tools/_motif.py:127, in motif_enrichment(motifs, regions, genome_fasta, backgroun
d, method)
    125 for key in result.keys():
    126     result[key]['adjusted p-value'] = _p_adjust_bh(result[key]['p-value'])
--> 127     result[key] = pl.DataFrame(result[key])
    128 return result

File ~/.conda/envs/ds_notebook/lib/python3.12/site-packages/polars/dataframe/frame.py:360, in DataFrame.__init__(self, data, schema, schema_overrides,
strict, orient, infer_schema_length, nan_to_null)
    355     self._df = dict_to_pydf(
    356         {}, schema=schema, schema_overrides=schema_overrides
    357     )
    359 elif isinstance(data, dict):
--> 360     self._df = dict_to_pydf(
    361         data,
    362         schema=schema,
    363         schema_overrides=schema_overrides,
    364         strict=strict,
    365         nan_to_null=nan_to_null,
    366     )
    368 elif isinstance(data, (list, tuple, Sequence)):
    369     self._df = sequence_to_pydf(
    370         data,
    371         schema=schema,
   (...)
    375         infer_schema_length=infer_schema_length,
    376     )

File ~/.conda/envs/ds_notebook/lib/python3.12/site-packages/polars/_utils/construction/dataframe.py:159, in dict_to_pydf(data, schema, schema_override
s, strict, nan_to_null, allow_multithreaded)
    146     data_series = [
    147         pl.Series(
    148             name,
   (...)
    154         for name in column_names
    155     ]
    156 else:
    157     data_series = [
    158         s._s
--> 159         for s in _expand_dict_values(
    160             data,
    161             schema_overrides=schema_overrides,
    162             strict=strict,
    163             nan_to_null=nan_to_null,
    164         ).values()
    165     ]
    167 data_series = _handle_columns_arg(data_series, columns=column_names, from_dict=True)
    168 pydf = PyDataFrame(data_series)

File ~/.conda/envs/ds_notebook/lib/python3.12/site-packages/polars/_utils/construction/dataframe.py:388, in _expand_dict_values(data, schema_overrides,
strict, order, nan_to_null)
    385     updated_data[name] = s
    387 elif arrlen(val) is not None or _is_generator(val):
--> 388     updated_data[name] = pl.Series(
    389         name=name,
    390         values=val,
    391         dtype=dtype,
    392         strict=strict,
    393         nan_to_null=nan_to_null,
    394     )
    395 elif val is None or isinstance(  # type: ignore[redundant-expr]
    396     val, (int, float, str, bool, date, datetime, time, timedelta)
    397 ):
    398     updated_data[name] = F.repeat(
    399         val, array_len, dtype=dtype, eager=True
    400     ).alias(name)

File ~/.conda/envs/ds_notebook/lib/python3.12/site-packages/polars/series/series.py:288, in Series.__init__(self, name, values, dtype, strict, nan_to_n
ull)
    285         raise TypeError(msg)
    287 if isinstance(values, Sequence):
--> 288     self._s = sequence_to_pyseries(
    289         name,
    290         values,
    291         dtype=dtype,
    292         strict=strict,
    293         nan_to_null=nan_to_null,
    294     )
    296 elif values is None:
    297     self._s = sequence_to_pyseries(name, [], dtype=dtype)

File ~/.conda/envs/ds_notebook/lib/python3.12/site-packages/polars/_utils/construction/series.py:294, in sequence_to_pyseries(name, values, dtype, stri
ct, nan_to_null)
    291     except RuntimeError:
    292         return PySeries.new_from_any_values(name, values, strict=strict)
--> 294 return _construct_series_with_fallbacks(
    295     constructor, name, values, dtype, strict=strict
    296 )

File ~/.conda/envs/ds_notebook/lib/python3.12/site-packages/polars/_utils/construction/series.py:312, in _construct_series_with_fallbacks(constructor,
name, values, dtype, strict)
    310 except TypeError:
    311     if dtype is None:
--> 312         return PySeries.new_from_any_values(name, values, strict=strict)
    313     else:
    314         return PySeries.new_from_any_values_and_dtype(
    315             name, values, dtype, strict=strict
    316         )

TypeError: unexpected value while building Series of type Int64; found value of type Float64: 0.0

Hint: Try setting `strict=False` to allow passing data with mixed types.
```