MOSDBDiscrete

In this module, we will have a brief overview of the `MOSDBDiscrete` class, which manages a transistor characterization database and provide methods for designers to query transistor small signal parameters.

MOSDBDiscrete example

To use the transistor characterization database, evaluate the following cell, which defines two methods, `query()` and `plot_data()`.

In [1]:
```
%matplotlib inline

import os
import pprint

import numpy as np
import matplotlib.pyplot as plt
# noinspection PyUnresolvedReferences
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib import ticker

from verification_ec.mos.query import MOSDBDiscrete

interp_method = 'spline'
spec_file = os.path.join(os.environ['BAG_WORK_DIR'], 'demo_data', 'mos_char_n
env_default = 'tt'
intent = 'standard'


def query(vgs=None, vds=None, vbs=0.0, vstar=None, env_list=None):
    """Get interpolation function and plot/query."""

    spec_list = [spec_file]
    if env_list is None:
        env_list = [env_default]

    # initialize transistor database from simulation data
    nch_db = MOSDBDiscrete(spec_list, interp_method=interp_method)
    # set process corners
    nch_db.env_list = env_list
    # set layout parameters
    nch_db.set_dsn_params(intent=intent)
    # returns a dictionary of smal-signal parameters
    return nch_db.query(vbs=vbs, vds=vds, vgs=vgs, vstar=vstar)


def plot_data(name='ibias', bounds=None, unit_val=None, unit_label=None):
    """Get interpolation function and plot/query."""
    env_list = [env_default]
    vbs = 0.0
    nvds = 41
    nvgs = 81
    spec_list = [spec_file]

    print('create transistor database')
    nch_db = MOSDBDiscrete(spec_list, interp_method=interp_method)
    nch_db.env_list = env_list
    nch_db.set_dsn_params(intent=intent)

    f = nch_db.get_function(name)
    vds_min, vds_max = f.get_input_range(1)
    vgs_min, vgs_max = f.get_input_range(2)
    if bounds is not None:
        if 'vgs' in bounds:
            v0, v1 = bounds['vgs']
            if v0 is not None:
                vgs_min = max(vgs_min, v0)
            if v1 is not None:
                vgs_max = min(vgs_max, v1)
        if 'vds' in bounds:
            v0, v1 = bounds['vds']
            if v0 is not None:
```

Querying Small-Signal Parameters

To lookup transistor small signal parameters given a bias point, use the `query()` method by evaluating the following cell. Feel free to play around with the numbers.

In [2]:
```
query(vgs=0.4, vds=0.5, vbs=0.0)
```

Out[2]:
```
{'cdb': 6.248976739750922e-17,
 'cdd': 2.0328230225209543e-16,
 'cds': -2.4163000626635453e-17,
 'cgb': 9.966702597590937e-19,
 'cgd': 1.6495553548122168e-16,
 'cgg': 3.6228642234598553e-16,
 'cgs': 1.9633421660500474e-16,
 'csb': 1.1021134465725374e-16,
 'css': 2.82382560635623e-16,
 'gb': 1.983603067386341e-05,
 'gds': 4.719944723025589e-06,
 'gm': 9.49214016617884e-05,
 'ibias': 1.2299113540770929e-05,
 'vstar': 0.2591431084128641 4,
 'vgs': 0.4,
 'vds': 0.5,
 'vbs': 0.0}
```

Plotting Small-Signal Parameters

`MOSDBDiscrete` stores each small signal parameter as a continuous function interpolated from simulation data. This makes it easy to manipulate those functions directly (such as using an optimization solver). For a simple example, the `plot_data()` method simply plots the functions versus V_{gs} and V_{ds}. Evaluate the following cell to see plots of various different small signal parameters.

In [5]:
```
%matplotlib inline

plot_data(name='ibias')
```

```
create transistor database
{'cdb': 6.40259557187212e-17,
 'cdd': 2.0379145885397679e-16,
 'cds': -3.5466715265482257e-17,
 'cgb': 1.5429673845401091e-18,
 'cgd': 1.7523221840073783e-16,
 'cgg': 3.827084324971782e-16,
 'cgs': 2.0593324671190025e-16,
 'csb': 1.0744529426477165e-16,
 'css': 2.7791182571118964e-16,
 'gb': 3.648119000000001e-05,
 'gds': 7.598960000000004e-06,
 'gm': 0.00016568942500000004,
 'ibias': 3.931712000000002e-05,
 'vbs': 0.0,
 'vds': 0.5025,
 'vgs': 0.6040000000000001,
 'vstar': 0.4745881639700302}
```