CaPS user manual

Release 0.5

Michael Hartmann, Gert-Ludwig Ingold

Nov 17, 2019

CONTENTS

CHAPTER

ONE

OVERVIEW AND FEATURES

CaPS provides software to describe the Casimir effect in the plane-sphere geometry for arbitrary temperatures and arbitrary non-magnetic materials constituting sphere and plate. Both objects are assumed to be in vacuum. The plane-sphere geometry is sketched in the inset of Figure 1.1 and is characterized by the sphere radius R and the distance L between sphere and infinite plane.

The main goal of the library and the associated programs is to compute the free energy $\mathcal{F}$ as a function of the radius R of the sphere, the separation L between sphere and plate, the temperature T, and the dielectric properties of the sphere and the plane. The code is highly optimized and – depending on parameters and the available resources – allows to compute the free energy for aspect ratios up to $R/L \sim 5\,000$. An idea of typical aspect ratios used in Casimir experiments in the plane-sphere geometry can be obtained from Figure 1.1.

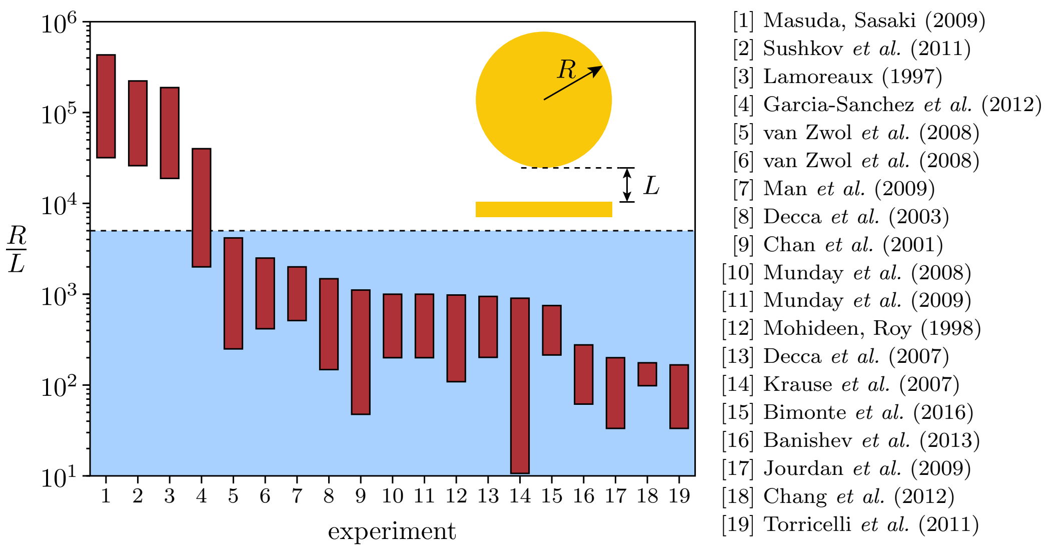

Figure 1.1: The aspect ratio R/L used in experiments in the plane-sphere geometry is shown by the red stripes. The blue area indicates the aspect ratios that are accessible using CaPS. The inset depicts the plane-sphere geometry where a sphere of radius R is placed at a distance L from a plane.

1.1 Features

CaPS provides the following main features:

- Computation of the free energy for aspect ratios used in typical experiments.
- Full support for perfect reflectors, metals described by the Drude and plasma model, and generic materials described by a user-defined dielectric function.
- Support for parallelization using MPI.
- Computation of the free energy in the high-temperature limit for perfect reflectors and metals described by the Drude or plasma model.

The computation of the high-temperature limit for the Drude model is based on G. Bimonte, T. Emig, "Exact Results for Classical Casimir Interactions: Dirichlet and Drude Model in the Sphere-Sphere and Sphere-Plane Geometry", Phys. Rev. Lett. 109, 160403 (2012).

Basic support for further geometries is provided for the special case of zero temperature and perfect reflectors:

- Computation of the free energy in the plane-cylinder geometry.
- Computation of the free energy for two spheres with equal radii.

The implementation for the plane-cylinder geometry is based on a symmetrized version of the matrix elements given in T. Emig, R. L. Jaffe, M. Kardar, and A. Scardicchio, "Casimir Interaction between a Plate and a Cylinder", Phys. Rev. Lett. 96, 080403 (2006).

1.2 Further reading

Some of the numerical ideas used in this library are described in M. Hartmann, G.-L. Ingold, P. A. Maia Neto, "Advancing numerics for the Casimir effect to experimentally relevant aspect ratios", Phys. Scr. 93, 114003 (2018). A more detailed description can be found in M. Hartmann, "Casimir effect in the plane-sphere geometry: Beyond the proximity force approximation", Ph.D. thesis (Universität Augsburg, 2018).

CHAPTER

TWO

INSTALLATION

In the following, we assume the operating system to be Ubuntu 18.04. The commands should also work on other Debian-like systems.

2.1 Compilation

The easiest way to obtain the source code of the CaPS package is by cloning it from Github. If not already present, install git by running

```
$ sudo apt install git
```

in a terminal. Here, the dollar sign indicates the shell prompt. Once git is installed, the command

```
$ git clone https://github.com/michael-hartmann/caps.git
```

will download the complete CaPS repository and store it in the directory `caps/`. As an alternative, you can also download and extract the zip- or tar.gz-archive of the latest release by navigating to `https://github.com/michael-hartmann/caps/releases`.

The CaPS library and the programs are written in C and C++ using LAPACK and MPI. The requirements to compile the source code are

- a C and C++ compiler,
- the development files for LAPACK and MPI,
- the build tools make and cmake.

These dependencies can be installed with:

```
$ sudo apt install gcc g++ libc6-dev libc++-dev cmake make libopenmpi-dev openmpi-
↪bin liblapack-dev
```

In order to compile the code, create a directory `bin` in the `caps/` directory and run `cmake` followed by `make`:

```
$ cd caps/
$ mkdir bin
$ cd bin/
$ cmake ..
$ make
```

The last command compiles the HODLR library, the libcaps library, and builds the shared objects `libhodlr.so` and `libcaps.so`. Then, the programs `caps` and `caps_logdetD` are built.

Note that alternative compilers can be specified by setting the variables `CC` and `CXX`. For the Intel C and C++ compilers, the last two commands above should be replaced by:

```
$ CC=icc CXX=icpc cmake ..
$ make
```

In order to run the programs, the system must be able to find the libraries `libhodlr.so` and `libcaps.so`. As the corresponding directories are not in the default search path, they need to be added to `LD_LIBRARY_PATH`

```
$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/hendrik/caps/bin
```

where we have assumed that the CaPS repository is in the directory `/home/hendrik`[1] .

After the previous steps, the programs `capc` (plane-cylinder geometry) and `cass` (sphere-sphere geometry with equal radii) to compute the Casimir free energy for perfect reflectors at $T = 0$ can be built with:

```
$ make capc
$ make cass
```

Under Ubuntu 18.10 we encountered problems linking to OpenBLAS resulting in error messages similar to:

```
undefined reference to 'dgemm_'
undefined reference to 'dstemr_'
undefined reference to 'dpotrf_'
undefined reference to 'dgemm_'
undefined reference to 'dgetrf_'
undefined reference to 'dgeqrf_'
undefined reference to 'ddot_'
```

In this case, we recommend using Atlas as a BLAS implementation. Make sure that Atlas is installed

```
$ sudo apt install libatlas-dev libatlas3-base
```

and compile the code using:

```
$ cd caps/
$ mkdir bin
$ cd bin/
$ cmake .. -DBLA_VENDOR=ATLAS
$ make
```

2.2 Testing

In order to verify that the compilation was successful, build and run the unit tests in `bin/`:

```
$ make tests
$ ./caps_tests
```

All tests should pass. Running the tests takes (depending on your hardware) about 7 minutes.

2.3 Improving performance

In order to improve performance, it might be necessary to tweak some compiler options. By default, the optimization level is `-O3` which usually yields reasonable performance.

If you either run and compile the code on the same machine, or the target machine supports the same instruction set, the option `-march=native` may increase performance. On an Intel Core i7-2600 machine, a performance boost of about 5% was found. To compile the code with this option, run:

```
$ cmake .. -DOPT="-O3 -march=native"
```

Similarly, link time optimization `-flto` might also increase performance. However, a test on an Intel Core i7-2600 machine showed basically no performance gain.

[1] In honor of Hendrik Brugt Gerhard Casimir (1909-2000).

CHAPTER

THREE

PROGRAMS

3.1 caps

The program `caps` computes the Casimir free energy $\mathcal{F}$ for the plane-sphere geometry as a sum

$$\mathcal{F} = \frac{k_\mathrm{B}T}{2} \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} \log\det\left(1 - \mathcal{M}^{(m)}(\xi_n)\right) \tag{3.1}$$

over the Matsubara frequencies $\xi_n = 2\pi n k_\mathrm{B} T/\hbar$. For zero temperature $T = 0$, the sum over the Matsubara frequencies becomes an integration. $\mathcal{M}^{(m)}$ denotes the round-trip operator associated with the scattering of an electromagnetic wave propagating from the sphere to the plane and back. Due to the axial symmetry of the plane-sphere geometry, in the multipole basis the round-trip operator becomes block-diagonal in the eigenvalues m of the z-component of the angular momentum.

The program supports a wide variety of options. A summary of all options can be obtained with `caps --help`. By default, the temperature is set to $T = 0$, and the sphere and plane are assumed to be perfect reflectors.

Please note that due to parallelization, the number of terms computed in the summation over m may vary from run to run. As a consequence, the numerical value obtained for the free energy also may vary from run to run while respecting the prescribed relative error. The abort criterion for the summation over m can be changed by the option `--cutoff` described in more detail below.

3.1.1 Mandatory options

There are two mandatory parameters:

- separation L between sphere and plane
- radius R of the sphere.

The program expects the lengths to be given in units of meters. As an example, the following command computes the Casimir interaction at $T = 0$ for perfect reflectors for a sphere of radius $R = 50\mu\mathrm{m}$ and a separation $L = 500\,\mathrm{nm}$:

```
$ mpirun -n 8 ./caps -R 50e-6 -L 500e-9
```

The command `mpirun` will set up the environment for MPI and the flag `-n` specifies how many processes the program should use. The first process is the master process. It delegates works to the other minion processes and collects the results once they are available. Due to this design, `caps` needs at least two processes to be able to start. As the master process does not consume much CPU time, set `-n` to N+1 to fully utilize the computational power of a computer with N processor cores.

The output of the above command looks similar to:

```
$ mpirun -n 8 ./caps -R 50e-6 -L 500e-9
# version: 0.5
# compiler: gcc
# compile time: Feb 25 2019 14:03:27
# compiled on: Linux host.name 4.9.0-8-amd64 x86_64
# pid: 19299
# start time: Tue Feb 26 09:53:35 2019
#
# LbyR = 0.01
# RbyL = 100
# L = 5e-07
# R = 5e-05
# T = 0
# cutoff = 1e-09
# epsrel = 1e-06
# iepsrel = 1e-08
# ldim = 701
# cores = 8
# quad = adaptive Gauss-Kronrod
#
# xi*(L+R)/c=50.5, logdetD=-9.60686704239278, t=16.8099
# xi*(L+R)/c=11769.7910188424, logdetD=-5.19555884457593e-101, t=3.0027
# xi*(L+R)/c=0.216677594013123, logdetD=-27.8461616612113, t=14.3809
# xi*(L+R)/c=1934.09140996997, logdetD=-5.35151875501729e-16, t=5.90594
# xi*(L+R)/c=1.31857780188353, logdetD=-27.4908297269515, t=15.1216
   ...
# xi*(L+R)/c=1.9476256940541, logdetD=-27.201610975434, t=14.9387
# xi*(L+R)/c=4.12332703671369, logdetD=-26.061368385508, t=14.979
# xi*(L+R)/c=2.63068289632324, logdetD=-26.8583901887955, t=14.8546
#
# ier=0, integral=-26.6608254093275, neval=165, epsrel=4.22793e-07
#
# 13525 determinants computed
# stop time: Tue Feb 26 10:36:20 2019
#
# L/R, L, R, T, ldim, E*(L+R)/(hbar*c)
0.00999999999999998, 5e-07, 5e-05, 0, 701, -428.5634172312517
```

The output adopts the CSV format and additional comments start with a number sign (#). The first comment section contains information on the compilation like the version of CaPS, time of compilation, name of compiler, machine where it was compiled and so on. A second comment section gives information about the geometry (radius R, minimal separation L, aspect ratio R/L, and inverse aspect ratio L/R) as well as numerical parameters and the employed integration technique (cutoff, epsrel, iepsrel, ldim, cores, quad). We will discuss the latter in more detail below. The value of the cores parameter is the number of MPI processes that were used for the computation.

The following section lists the numerical results for the determinant of the scattering matrix printed for the different Matsubara frequencies at which it was evaluated. The comment starting with `ier` gives the result of the integration. If the integration was successful, the value is `ier=0`, see also the description of dqags of QUADPACK. The program ends by printing the result of the computation. The free energy is given in units of $(L+R)/\hbar c$. For the present example the free energy is

$$\mathcal{F} \approx \frac{-428.6\hbar c}{50\mu\mathrm{m} + 500\mathrm{nm}} \approx -2.68 \times 10^{-19}\mathrm{J}.$$

The PFA result in this case is $\mathcal{F}_{\mathrm{PFA}} = \hbar c \pi^3 R/720 L^2 \approx -2.72 \times 10^{-19}$J.

The desired relative accuracy of the integration over the Matsubara frequencies can be set using `--epsrel`. By default, `EPSREL` is 10^{-6}. Note that the integrand needs to be sufficiently smooth. In particular, for very low values of `EPSREL` you might need to decrease the value of `CUTOFF` using `--cutoff`. The value of `CUTOFF` determines when the summation over m is stopped. The abort criterion is:

$$\frac{\log\det\left(1-\mathcal{M}^{(m)}(\xi)\right)}{\log\det\left(1-\mathcal{M}^{(0)}(\xi)\right)} < \mathrm{CUTOFF}$$

The default value of `CUTOFF` is 10^{-9}. As a rule of thumb, in order for the integrand to be sufficiently smooth for the integration routine, `CUTOFF` should be at least two orders of magnitude smaller than `EPSREL`.

By default, the integration routine uses an adaptive Gauss-Kronrod method provided by CQUADPACK. For perfect reflectors it is sometimes faster to use an adaptive exponentially convergent Fourier-Chebyshev quadrature scheme (FCQS), see J. P. Boyd, "Exponentially convergent Fourier-Chebshev quadrature schemes on bounded and infinite intervals", J. Sci. Comput. 2, 99 (1987). FCQS can be enabled by the flag `--fcqs`. Since the adaptive algorithm for FCQS is not well tested, this option is considered experimental. Moreover, it is not recommended to use FCQS for any other materials than perfect reflectors.

3.1.2 Temperature

The temperature in Kelvin can be set by using the flag `-T`. The following call computes the free energy just like in the last example but at room temperature $T = 300\mathrm{K}$:

```
$ mpirun -n 8 ./caps -R 50e-6 -L 500e-9 -T 300
# version: 0.5
# compiler: gcc
# compile time: Feb 25 2019 14:03:27
# compiled on: Linux host.name 4.9.0-8-amd64 x86_64
# pid: 20396
# start time: Tue Feb 26 11:25:28 2019
#
# LbyR = 0.01
# RbyL = 100
# L = 5e-07
# R = 5e-05
# T = 300
# using Matsubara spectrum decomposition (MSD)
# cutoff = 1e-09
# epsrel = 1e-06
# iepsrel = 1e-08
# ldim = 701
# cores = 8
# model = perfect reflectors
#
# xi*(L+R)/c=0, logdetD=-27.8619907991614, t=0.193124
# xi*(L+R)/c=41.5698805095683, logdetD=-11.5794516487266, t=16.6413
# xi*(L+R)/c=83.1397610191367, logdetD=-4.91948448017641, t=17.6647
# xi*(L+R)/c=124.709641528705, logdetD=-2.13175536619868, t=18.3457
# xi*(L+R)/c=166.279522038273, logdetD=-0.930897479022303, t=19.1386
# xi*(L+R)/c=207.849402547842, logdetD=-0.407803084237451, t=19.1952
# xi*(L+R)/c=249.41928305741, logdetD=-0.178887508701769, t=19.4133
# xi*(L+R)/c=290.989163566978, logdetD=-0.0785144339556277, t=19.431
# xi*(L+R)/c=332.559044076547, logdetD=-0.0344676750184605, t=19.1501
# xi*(L+R)/c=374.128924586115, logdetD=-0.015132238792835, t=19.1419
# xi*(L+R)/c=415.698805095683, logdetD=-0.00664345573274249, t=19.0193
# xi*(L+R)/c=457.268685605252, logdetD=-0.00291655594485503, t=18.7029
# xi*(L+R)/c=498.83856611482, logdetD=-0.00128033535183925, t=18.4298
# xi*(L+R)/c=540.408446624388, logdetD=-0.000562015614133802, t=17.8942
# xi*(L+R)/c=581.978327133957, logdetD=-0.000246682939573313, t=17.5315
# xi*(L+R)/c=623.548207643525, logdetD=-0.000108265748892663, t=17.0982
# xi*(L+R)/c=665.118088153093, logdetD=-4.75115918214971e-05, t=16.4634
# xi*(L+R)/c=706.687968662662, logdetD=-2.08477925150647e-05, t=15.5548
#
# 1686 determinants computed
# stop time: Tue Feb 26 11:30:37 2019
#
# L/R, L, R, T, ldim, E*(L+R)/(hbar*c)
0.00999999999999998, 5e-07, 5e-05, 300, 701, -452.7922092119535
```

For finite temperatures, the free energy is no longer given as an integral, but as the sum (3.1) over Matsubara frequencies ξ_n. The summation over n is stopped once

$$\frac{\log\det(1-\mathcal{M}(\xi_n))}{\log\det(1-\mathcal{M}(0))} < \text{EPSREL}.$$

By default, `EPSREL` is 10^{-6}. Its value can be modified by means of the option `--epsrel`.

By default, the free energy is computed by means of (3.1) referred to as Matsubara spectrum decomposition (MSD). An alternative approach is the Padé spectrum decomposition (PSD). PSD is an optimal sum-over-poles expansion scheme explained in J. Hu, M. Luo, F. Jiang, R.-X. Xu, Y. J. Yan, "Padé spectrum decompositions of quantum distribution functions and optimal hierarchical equations of motion construction for quantum open systems", J. Chem. Phys. 134, 244106 (2010). The PSD requires the evaluation of fewer terms as compared to the MSD. PSD can be enabled with the flag `--psd`. The order is determined automatically to achieve a relative error of the order specified by `--epsrel`, but can also be set manually using `--psd-order`. Since the automatic determination of the order is not well tested, PSD is considered experimental.

The high-temperature limit of the Casimir free energy requires only the evaluation of $\log\det(1-\mathcal{M}(0))$ and can be obtained by means of the flag `--ht`. It will be determined for Drude metals and perfect reflectors. In contrast to the cases of zero and finite temperatures, the result is given in units of $k_\mathrm{B}T$. While for perfect reflectors, no analytical result is known, for Drude metals the analytical formula given in G. Bimonte, T. Emig, PRL 109, 160403 (2012) is used.

For example, for a sphere of radius $R = 100\mu\mathrm{m}$ and a smallest separation $L = 100\mathrm{nm}$ one finds:

```
$ mpirun -n 8 ./caps -R 100e-6 -L 100e-9 --ht
# version: 0.5
# compiler: gcc
# compile time: Feb 26 2019 17:42:26
# compiled on: Linux host.name 4.9.0-8-amd64 x86_64
# pid: 29426
# start time: Tue Feb 26 17:42:50 2019
#
# LbyR = 0.001
# RbyL = 1000
# L = 1e-07
# R = 0.0001
# high-temperature limit
# cutoff = 1e-09
# epsrel = 1e-06
# iepsrel = 1e-08
# ldim = 7001
# cores = 8
#
# L/R, L, R, ldim, E_Drude/(kB*T), E_PR/(kB*T)
0.0009999999999999998, 1e-07, 0.0001, 7001, -149.6981411829862, -296.4343145093178
```

Compared to zero or finite temperatures, it is considerably less demanding to compute the high-temperature limit of the Casimir free energy. For the above example, the aspect ratio is $R/L = 1000$, but the computation time on a standard desktop computer using 8 cores is only about 13 seconds.

3.1.3 Material parameters

The examples presented so far were mostly for sphere and plate made of perfect reflectors. In addition, it is possible to do calculations for the plasma model with the imaginary-frequency dielectric function

$$\epsilon(\mathrm{i}\xi) = 1 + \frac{\omega_\mathrm{P}^2}{\xi^2}, \tag{3.2}$$

for the Drude model with

$$\epsilon(\mathrm{i}\xi) = 1 + \frac{\omega_\mathrm{P}^2}{\xi(\xi+\gamma)}, \tag{3.3}$$

and for materials with a user-defined dielectric function. Both objects, sphere and plate, are assumed to consist of the same material.

The plasma model allows to account for high-frequency transparency of the material and is characterized by the plasma frequency ω_P appearing in (3.2). This parameter can be set using `--omegap`. and its value is expected to be given in units of $\mathrm{eV}/\hbar$. For example, for $R = 50\mu\mathrm{m}$, $L = 500\mathrm{nm}$, $T = 300\mathrm{K}$, and plasma frequency $\omega_\mathrm{P} = 9\mathrm{eV}/\hbar$ appropriate for gold, the Casimir free energy assuming the plasma model is:

```
$ mpirun -n 8 ./caps -R 50e-6 -L 500e-9 -T 300 --omegap 9
# version: 0.5
# compiler: gcc
# compile time: Feb 25 2019 14:03:27
# compiled on: Linux host.name 4.9.0-8-amd64 x86_64
# pid: 20511
# start time: Tue Feb 26 11:31:38 2019
#
# LbyR = 0.01
# RbyL = 100
# L = 5e-07
# R = 5e-05
# T = 300
# using Matsubara spectrum decomposition (MSD)
# cutoff = 1e-09
# epsrel = 1e-06
# iepsrel = 1e-08
# ldim = 701
# cores = 8
# omegap = 9
# gamma = 0
# model = plasma
#
# xi*(L+R)/c=0, logdetD=-26.6976344223495, t=1.0548
# xi*(L+R)/c=41.5698805095683, logdetD=-10.4687593311701, t=29.2239
# xi*(L+R)/c=83.1397610191367, logdetD=-4.17127445011799, t=30.5278
# xi*(L+R)/c=124.709641528705, logdetD=-1.69110879470416, t=31.9545
# xi*(L+R)/c=166.279522038273, logdetD=-0.68977177243457, t=32.2149
# xi*(L+R)/c=207.849402547842, logdetD=-0.28194807348852, t=33.1459
# xi*(L+R)/c=249.41928305741, logdetD=-0.115330010733512, t=33.0782
# xi*(L+R)/c=290.989163566978, logdetD=-0.0471851591737746, t=33.3165
# xi*(L+R)/c=332.559044076547, logdetD=-0.0193059198900916, t=36.3263
# xi*(L+R)/c=374.128924586115, logdetD=-0.00789924565082466, t=36.4202
# xi*(L+R)/c=415.698805095683, logdetD=-0.00323219686668301, t=33.851
# xi*(L+R)/c=457.268685605252, logdetD=-0.00132263206346751, t=31.1036
# xi*(L+R)/c=498.83856611482, logdetD=-0.000541279202385481, t=30.3951
# xi*(L+R)/c=540.408446624388, logdetD=-0.000221541550914926, t=29.6892
# xi*(L+R)/c=581.978327133957, logdetD=-9.06879127562377e-05, t=28.5112
# xi*(L+R)/c=623.548207643525, logdetD=-3.71288341482914e-05, t=27.586
# xi*(L+R)/c=665.118088153093, logdetD=-1.5203622898795e-05, t=26.199
#
# 1582 determinants computed
# stop time: Tue Feb 26 11:40:03 2019
#
# L/R, L, R, T, ldim, E*(L+R)/(hbar*c)
0.00999999999999998, 5e-07, 5e-05, 300, 701, -408.1688659974158
```

The Drude model (3.3) not only accounts for the high-frequency cutoff but also a finite zero-frequency conductivity $\sigma_0 = \omega_\mathrm{P}^2/\gamma$. The additional parameter γ can be specified by the flag `--gamma` with a value given in units of $\mathrm{eV}/\hbar$. For gold, $\gamma = 35\mathrm{meV}/\hbar$ and extending the previous example to the Drude model yields:

```
$ mpirun -n 8 ./caps -R 50e-6 -L 500e-9 -T 300 --omegap 9 --gamma 0.035
# version: 0.5
# compiler: gcc
# compile time: Feb 25 2019 14:03:27
# compiled on: Linux host.name 4.9.0-8-amd64 x86_64
# pid: 24336
# start time: Tue Feb 26 12:08:33 2019
#
# LbyR = 0.01
# RbyL = 100
# L = 5e-07
# R = 5e-05
# T = 300
# using Matsubara spectrum decomposition (MSD)
# cutoff = 1e-09
# epsrel = 1e-06
# iepsrel = 1e-08
# ldim = 701
# cores = 8
# omegap = 9
# gamma = 0.035
# model = drude
#
# xi*(L+R)/c=0, logdetD=-14.5697227167729, t=0.000477076
# xi*(L+R)/c=41.5698805095683, logdetD=-10.3653815649851, t=29.2174
# xi*(L+R)/c=83.1397610191367, logdetD=-4.13628095371182, t=30.4618
# xi*(L+R)/c=124.709641528705, logdetD=-1.67763657179774, t=31.9241
# xi*(L+R)/c=166.279522038273, logdetD=-0.684394806814442, t=32.5905
# xi*(L+R)/c=207.849402547842, logdetD=-0.279773855516853, t=33.1029
# xi*(L+R)/c=249.41928305741, logdetD=-0.114446241958121, t=33.6549
# xi*(L+R)/c=290.989163566978, logdetD=-0.0468251449212843, t=33.7789
# xi*(L+R)/c=332.559044076547, logdetD=-0.0191591315699105, t=32.8368
# xi*(L+R)/c=374.128924586115, logdetD=-0.00783937539816867, t=34.304
# xi*(L+R)/c=415.698805095683, logdetD=-0.00320777554704996, t=33.1737
# xi*(L+R)/c=457.268685605252, logdetD=-0.00131267076024208, t=31.7234
# xi*(L+R)/c=498.83856611482, logdetD=-0.000537216353887839, t=30.7169
# xi*(L+R)/c=540.408446624388, logdetD=-0.000219884619851423, t=30.7112
# xi*(L+R)/c=581.978327133957, logdetD=-9.00122403077102e-05, t=29.2664
# xi*(L+R)/c=623.548207643525, logdetD=-3.685333004516e-05, t=27.5843
# xi*(L+R)/c=665.118088153093, logdetD=-1.50912958983084e-05, t=26.3201
# xi*(L+R)/c=706.687968662662, logdetD=-6.18096543194596e-06, t=24.9774
#
# 1603 determinants computed
# stop time: Tue Feb 26 12:17:19 2019
#
# L/R, L, R, T, ldim, E*(L+R)/(hbar*c)
0.009999999999999998, 5e-07, 5e-05, 300, 701, -325.8011897578629
```

General materials can be defined by the user and specified by the flag `--material`. Its parameter should be a path to a file describing the material in the following format:

```
# Drude parameters for low frequencies
# omegap_low = 9.0eV
# gamma_low  = 0.03eV
#
# Drude parameters for high frequencies
# omegap_high = 54.475279eV
# gamma_high  = 211.48855eV
#
# xi in rad/s            epsilon(i*xi)
151900000000.0000        27202177.31278406
167090000000.0000        24722324.84701070
182280000000.0000        22655566.74077545
...
1.488620000000000E+018    1.002550948190463
1.503810000000000E+018    1.002504731170786
1.519000000000000E+018    1.002459773692494
```

Each line either starts with a number sign (#) or contains a frequency ξ in units of $\mathrm{rad/s}$ and the corresponding value of the dielectric function $\epsilon(\mathrm{i}\xi)$ separated by either tabs or spaces. The frequencies have to be in ascending order. The dielectric function for an arbitrary frequency is then computed using linear interpolation. For frequencies smaller than the smallest frequency provided in the file, the dielectric function is computed using the Drude model (3.3) with the plasma frequency given by `omegap_low` and the relaxation frequency given by `gamma_low`. If `omegap_low` and `gamma_low` are not given in the file, the dielectric function is assumed to be 1. The behavior for frequencies larger than the largest provided frequency is analogous using the parameters given by `omegap_high` and `gamma_high`. More details can be found in the directory `materials/`.

The following example computes the Casimir energy for a sphere of $R = 50\mu\mathrm{m}$ at separation $L = 500\mathrm{nm}$ at room temperature $T = 300\mathrm{K}$ for real gold:

```
$ mpirun -n 8 ./caps -R 50e-6 -L 500e-9 -T 300 --material ../materials/gold.csv
# version: 0.5
# compiler: gcc
# compile time: Feb 25 2019 14:03:27
# compiled on: Linux host.name 4.9.0-8-amd64 x86_64
# pid: 24642
# start time: Tue Feb 26 12:18:36 2019
#
# LbyR = 0.01
# RbyL = 100
# L = 5e-07
# R = 5e-05
# T = 300
# using Matsubara spectrum decomposition (MSD)
# cutoff = 1e-09
# epsrel = 1e-06
# iepsrel = 1e-08
# ldim = 701
# cores = 8
# filename = ../materials/gold.csv
# model = optical data (xi=0: Drude)
# plasma = -26.6976344217189 (logdetD(xi=0) for plasma model with omegap=9eV)
#
# xi*(L+R)/c=0, logdetD=-14.5697227167729, t=1.0434
# xi*(L+R)/c=41.5698805095683, logdetD=-10.3908796547603, t=29.5941
# xi*(L+R)/c=83.1397610191367, logdetD=-4.15625893810359, t=30.6949
# xi*(L+R)/c=124.709641528705, logdetD=-1.6919107999977, t=32.3372
# xi*(L+R)/c=166.279522038273, logdetD=-0.693653981851821, t=32.8614
# xi*(L+R)/c=207.849402547842, logdetD=-0.285378030697434, t=33.7248
# xi*(L+R)/c=249.41928305741, logdetD=-0.117672555842387, t=33.5534
# xi*(L+R)/c=290.989163566978, logdetD=-0.0486088504532428, t=33.1554
# xi*(L+R)/c=332.559044076547, logdetD=-0.0201151891681236, t=33.0375
# xi*(L+R)/c=374.128924586115, logdetD=-0.00833817561811858, t=32.8751
# xi*(L+R)/c=415.698805095683, logdetD=-0.00346241156603652, t=32.5475
# xi*(L+R)/c=457.268685605252, logdetD=-0.00144011362577151, t=31.6877
# xi*(L+R)/c=498.83856611482, logdetD=-0.000600031780813664, t=31.1144
# xi*(L+R)/c=540.408446624388, logdetD=-0.000250400886432969, t=31.9239
# xi*(L+R)/c=581.978327133957, logdetD=-0.000104657168106622, t=31.2511
# xi*(L+R)/c=623.548207643525, logdetD=-4.3804324287031e-05, t=28.8938
# xi*(L+R)/c=665.118088153093, logdetD=-1.83601183587417e-05, t=27.7136
# xi*(L+R)/c=706.687968662662, logdetD=-7.70510781683187e-06, t=26.0717
#
# 1681 determinants computed
# stop time: Tue Feb 26 12:27:30 2019
#
# L/R, L, R, T, ldim, E*(L+R)/(hbar*c)
0.00999999999999998, 5e-07, 5e-05, 300, 701, -326.8806691538941
```

As indicated in the comment line referring to the model used, the zeroth Matsubara frequency is evaluated by means of a Drude model. In order to obtain the corresponding result where the plasma model is used instead for the zeroth Matsubara frequency, the value given in the comment line starting with `# plasma =` can be used. The value given there, i.e. -26.69... in our example, is given in units of $k_\mathrm{B}T/2$ and corresponds to the additional contribution in the high-temperature limit to the energy in the plasma model. In our example, the free energy using the Drude model at zero frequency is

$$\mathcal{F}_\mathrm{Drude} \approx -326.88\frac{\hbar c}{L+R} \approx -2.0464\times 10^{-19}\mathrm{J},$$

while assuming the plasma model for zero frequency we find

$$\mathcal{F}_\mathrm{plasma} \approx \mathcal{F}_\mathrm{Drude} - 26.69763\frac{k_\mathrm{B}T}{2} \approx -2.5993\times 10^{-19}\mathrm{J}.$$

3.1.4 Truncation of the vector space

The required dimension of the vector space

$$\ell_\mathrm{dim} = \eta\frac{R}{L} \tag{3.4}$$

scales linearly with the aspect ratio R/L with a prefactor η related to the estimated relative error according to the following table:

relative error	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
η	2.8	4	5.2	6.4	7.6	8.8	10

The truncation of the vector space is discussed in more detail in M. Hartmann, "Casimir effect in the plane-sphere geometry: Beyond the proximity force approximation", PhD thesis (Universität Augsburg, 2018).

The dimension of the vector space can be specified by means of the flag `--ldim` fixing ℓ_dim. Alternatively, the flag `--eta` can be used from which the dimension is obtained according to

$$\ell_\mathrm{dim} = \max\left(20, \lceil \eta R/L\rceil\right)$$

where $\lceil x\rceil$ denotes the smallest integer larger than x. By default, the dimension of the vector space is determined from (3.4) with $\eta = 7$.

3.1.5 Other options

The computation of the matrix elements of the round-trip operator involves an integration. The desired relative error for this integration can be set using `--iepsrel`. The default value of 10^{-8} should be sufficient for almost all purposes. If the Casimir energy needs to be determined to very high accuracy with a relative error of 10^{-7} or smaller, it is recommended to decrease `IEPSREL` accordingly.

If `caps` was interrupted, e.g. when the time limit on a compute cluster was exceeded, the `--resume` option can be used to resume the computation on the basis of the partial output created so far. If it is given the option `--resume FILENAME`, `caps` reads the content of `FILENAME` and re-uses the computed values. It is the responsibility of the user to make sure that all other parameters given to `caps` exactly match the parameters used to generate `FILENAME` in a previous run.

3.2 caps_logdetD

The program `caps_logdetD` computes the building block of (3.1)

$$\log\det\left(1-\mathcal{M}^{(m)}(\xi)\right)$$

which, in addition to the parameters introduced above, depends on m and ξ. Accordingly, there exist two additional mandatory options besides `-L` and `-R`, namely `-m` and `--xi`. The frequency given by `--xi` is expected in units of $c/(L+R)$.

The options `-L`, `-R`, `--ldim`, `--material`, and `--iepsrel` are the same as described in Section 3.1 for the program `caps`. In addition, the algorithm used to compute the determinant can be specified with `--detalg`. Valid values are HODLR, QR, LU, and Cholesky.

A typical output looks like

```
$ ./caps_logdetD -R 100e-6 -L 1e-6 -m 1 --xi 1
# ./caps_logdetD, -R, 100e-6, -L, 1e-6, -m, 1, --xi, 1
# L/R    = 0.0099999999999999998
# L      = 1e-06
# R      = 0.0001
# ldim   = 501
# epsrel = 1.0e-08
# detalg = HODLR
#
# L, R, ξ*(L+R)/c, m, logdet(Id-M), ldim, time
1e-06, 0.0001, 1, 1, -6.463973151333013, 501, 0.548678
```

Sometimes, it is useful to dump the round-trip matrix in Numpy format. If the environment variable `CAPS_DUMP` is set and `detalg` is not HODLR, the round-trip matrix will be saved to the filename contained in `CAPS_DUMP`. Also note that if `detalg` is Cholesky, only the upper half of the matrix is computed.

The following example demonstrates how to generate and save a round-trip matrix:

```
$ CAPS_DUMP=M.npy ./caps_logdetD -R 100e-6 -L 1e-6 -m 1 --xi 2 --detalg LU
# ./caps_logdetD, -R, 100e-6, -L, 1e-6, -m, 1, --xi, 2, --detalg, LU
# L/R    = 0.0099999999999999998
# L      = 1e-06
# R      = 0.0001
# ldim   = 501
# epsrel = 1.0e-08
# detalg = LU
#
# L, R, ξ*(L+R)/c, m, logdet(Id-M), ldim, time
1e-06, 0.0001, 2, 1, -6.349155228127988, 501, 0.817265
```

The newly created file `M.npy` containing the NumPy dump of the round-trip matrix can be read back into a Python session as follows:

```
$ python
Python 3.6.5 | packaged by conda-forge | (default, Apr  6 2018, 13:39:56)
[GCC 4.8.2 20140120 (Red Hat 4.8.2-15)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> M = np.load("M.npy")    # load matrix
>>> dim,_ = M.shape         # get dimension
>>> Id = np.eye(dim)        # identity matrix
>>> np.linalg.slogdet(Id-M) # compute log of determinant
(1.0, -6.349155228128008)
```

3.3 capc

The program `capc` computes the Casimir interaction in the cylinder-plane geometry for perfect reflectors at zero temperature. The radius of the cylinder is specified by means of the flag `-R` and the smallest separation between cylinder and plate is specified by `-d`. Both lengths are expected to be given in meters. The length L of the cylinder is assumed to be large compared to the cylinder radius R and the cylinder-plate distance d so that the cylinder can be assumed to be infinitely long.

The following example computes the Casimir energy per unit length for an infinitely long cylinder of radius $R = 100\mu\mathrm{m}$ and a separation of $d = 100\mathrm{nm}$:

```
$ ./capc -R 100e-6 -d 100e-9
# R/d = 1000
# d = 1e-07
# R = 0.0001
# T = 0
# lmax = 6000
# epsrel = 1e-08
#
# d/R, d, R, T, lmax, E_PFA/(L*hbar*c), E_D/E_PFA, E_N/E_PFA, E_EM/E_PFA
0.001, 1e-07, 0.0001, 0, 6000, -72220981652413.5, 0.500089151078031, 0.
↪499432943796718, 0.999522094874749
```

E_D and E_N correspond to Dirichlet and Neumann boundary conditions, respectively, and E_EM is the energy for the electromagnetic field with E_EM = E_D + E_N. All results are given as ratios with respect to the free energy per unit length calculated within the proximity-force approximation.

A full list of options accepted by capc can be obtained by capc --help.

3.4 cass

The program cass computes the Casimir energy in the sphere-sphere geometry for perfect reflectors at zero temperature. It is assumed that both spheres have the same radius $R = R_1 = R_2$. The radius is specified by means of the flag -R and the smallest distance between the two spheres is specified by -d. Both lengths are expected to be given in units of meters.

The round-trip operator in the sphere-sphere geometry is given as

$$\mathcal{M}_{\mathrm{SS}} = \mathcal{R}_1\mathcal{T}_{12}\mathcal{R}_2\mathcal{T}_{21}\,.$$

Here, $\mathcal{R}_j$ denotes the reflection operator at sphere j, and $\mathcal{T}_{ij}$ is the translation operator from sphere j to sphere i. After symmetrization and for equal radii, the round-trip operator can be written as:

$$\widehat{\mathcal{M}}_{\mathrm{SS}} = \underbrace{\sqrt{\mathcal{R}}\mathcal{T}\sqrt{\mathcal{R}}}_{=A}\underbrace{\sqrt{\mathcal{R}}\mathcal{T}\sqrt{\mathcal{R}}}_{=A} = A^2\,.$$

Since for perfect reflectors the Fresnel coefficients are $r_{\mathrm{TM}} = -r_{\mathrm{TE}} = 1$, the operator A can be expressed as $A = \mathcal{M}_{\mathrm{PS}}$ where $\mathcal{M}_{\mathrm{PS}}$ is the symmetrized round-trip operator in the plane-sphere geometry. This idea basically amounts to using the method of image charges.

The following example computes the Casimir energy in the sphere-sphere geometry for $R_1 = R_2 = 100\mu\mathrm{m}$ and $d = 10\mu\mathrm{m}$:

```
$ ./cass -R 100e-6 -d 10e-6
# version: 0.5
# compiler: gcc
# compile time: Mar 13 2019 22:10:53
# compiled on: Linux host.name 4.9.0-8-amd64 x86_64
#
# sphere-sphere geometry
# model: perfect reflectors
# R1 = R2 = 0.0001
# d = 1e-05
# T = 0
# ldim = 50
# epsrel = 1e-06
# iepsrel = 1e-09
# cutoff = 1e-10
#
# xi*d/c=1, logdet=-0.303363189004
# xi*d/c=233.06516869, logdet=-3.13138917509e-204
# xi*d/c=0.004290645426, logdet=-1.68510617725
```

```
        ...
# xi*d/c=0.0385668454268, logdet=-1.65999004318
# xi*d/c=0.081650040331, logdet=-1.60322260646
# xi*d/c=0.0520927306203, logdet=-1.64443634206
#
# ier = 0, neval = 135
#
# R1, R2, L, E*d/(hbar*c), relative error (due to integration)
0.0001, 0.0001, 1e-05, -0.166155548334, 5.80089e-07
```

The runtime of this program is about 4 minutes. For a full list of options see `cass --help`.

The program `cass` carries out a full matrix-matrix multiplication using LAPACK to compute $\widehat{\mathcal{M}}_{\mathrm{SS}}$ and a LU decomposition to compute the determinant of the scattering matrix. Note that this program does not support parallelization. Therefore, the program is useful only for not too large aspect ratios R/d.

CHAPTER FOUR

API DOCUMENTATION

The documentation of the API is available at `manual/api.pdf` or can be generated running

```
$ doxygen doxygen.conf
```

in the directory `src/`. The documentation will be generated in `docs/api/`. You need doxygen installed on your computer.

CHAPTER

FIVE

EXAMPLES

In the directory `examples/` examples of simple programs can be found that demonstrate how to use the CaPS API. The examples are kept simple and well documented.