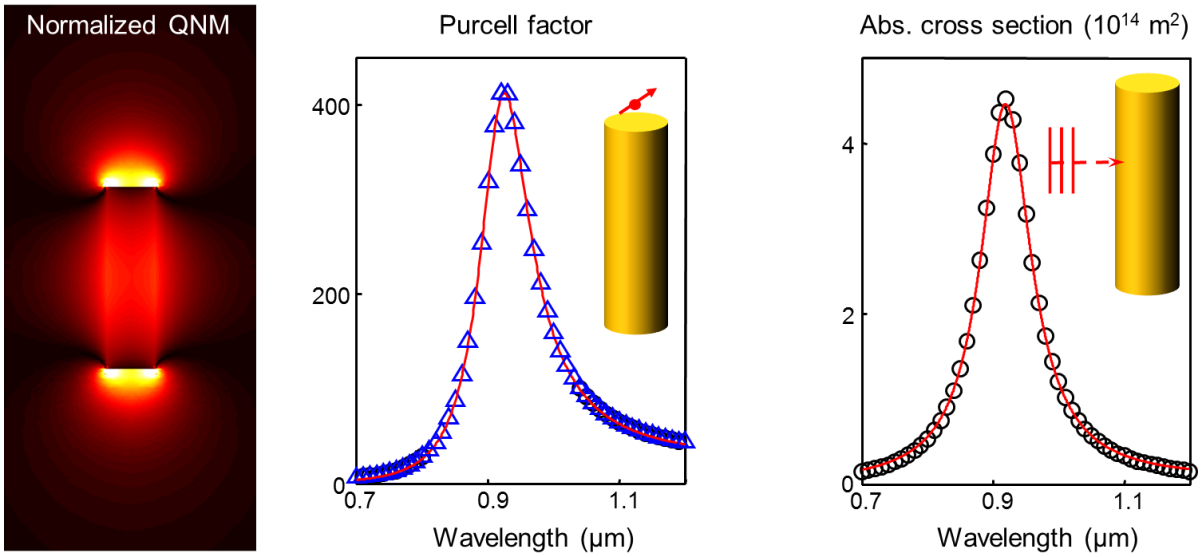
QNMPole: an open-source for calculating resonance modes of plasmonic nanoresonators and photonic microresonators (version 7)

Kevin Cognée, Jianji Yang, and Philippe Lalanne
LP2N, Institut d'Optique d'Aquitaine, IOGS, Univ. Bordeaux, CNRS.
jianji.yang.photonique@gmail.com & jjyang10@stanford.edu
kevin.cognee@institutoptique.fr
philippe.lalanne@institutoptique.fr

Last revision: November, 2019



QNMPole: an open-source for calculating resonance modes of plasmonic nanoresonators and photonic microresonators

The present documentation provides the user guide of **QNMPole**, which is a Matlab-based open-source numerical tool, built for calculating and normalizing the resonance modes (also called the quasi-normal modes or QNMs) of plasmonic and photonic resonators [1]. **QNMPole** can be used to calculate the absorption and scattering cross-sections of nanoresonators, either using QNM data or fully-vectorial data. A straightforward extension can be used very simply to calculate the Purcell factor of nanoresonators [1]. The open-source numerical tool relies on COMSOL-multiphysics, via Matlab-COMSOL livelink, to achieve fully automated calculation and normalization of the QNMs.

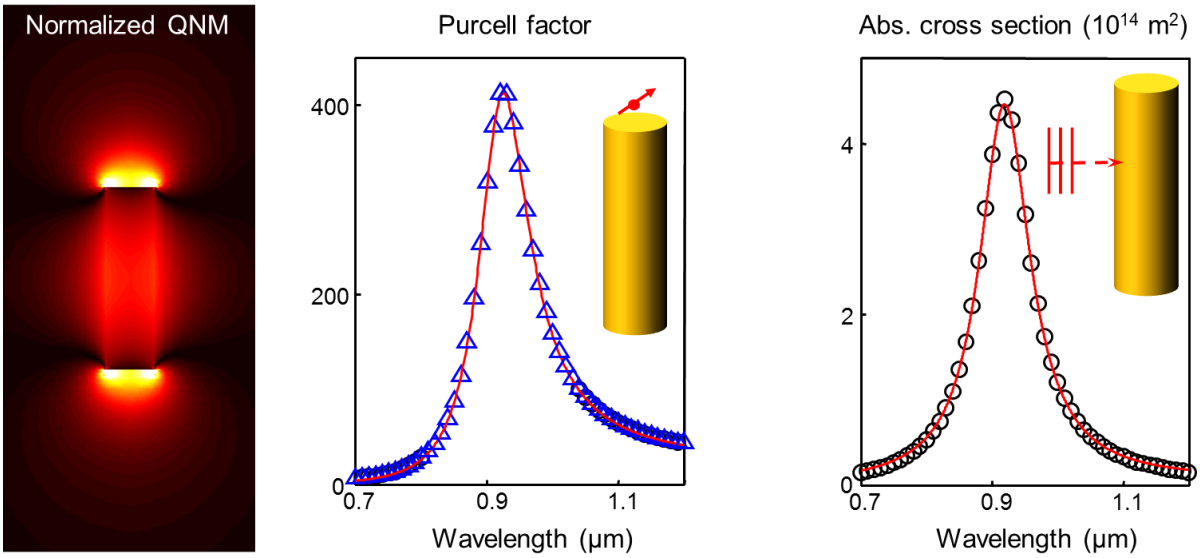


Figure 1. **Illustration of the retrieval procedure of the radiation diagram.** (**left**) Vertical electric-field component (absolute value) of the fundamental QNM of a gold nanorod. (**middle**) Purcell factor of a dipole source placed in the near-field of the nanorod; fully-vectorial data (blue triangle) are compared to analytical expressions (solid-red curve) computed from the normalized QNM field. (**right**) Absorption cross-section of the nanorod under plane-wave illumination; fully-vectorial data (black circles) are compared to analytical expressions (solid-red curve) computed from the normalized QNM field. After [1].

[1] Q. Bai, M. Perrin, C. Sauvan, J.P. Hugonin and P. Lalanne, "Efficient and intuitive method for the analysis of light scattering by a resonant nanostructure", Opt. Express 21, 27371-82 (2013)

Content

1. INTRODUCTION TO QNMPOLE

QNMPole calculates and normalizes (calculates the mode volume) of resonance modes (also called the quasi-normal modes or QNMs) of plasmonic and photonic resonators [1].

In this section, basic information about **QNMPole** is provided.

1.1 Download & Installation

QNMPole is a freeware that operates under Matlab environment coupled to a COMSOL environment via Matlab-COMSOL livelink. To install it, copy and decompress the companion folder "QNMPole.zip" and add the folder in the Matlab path.

1.2 How to acknowledge and cite

We kindly ask that you reference the **QNMPole** package from IOGS-CNRS and its authors in any publication/report for which you used it. The preferred citation for QNMPole is the following paper:

[ref] Q. Bai, M. Perrin, C. Sauvan, J.P. Hugonin and P. Lalanne, "Efficient and intuitive method for the analysis of light scattering by a resonant nanostructure", Opt. Express **21**, 27371-82 (2013).

A brief description of the algorithm might be:

"The resonance mode calculation and normalization was computed with a general iterative method based on a complex-frequency search using a freely available software package [ref]."

1.3 Units and conventions of input/output data for QNMPole

Unit. All the input information is required to be in the **SI unit** (e.g., volts per meter for electric field **E**, amperes per meter for magnetic field **H**, …). Accordingly, the output information is given in **SI unit** as well.
Convention. The time dependent terms exp(*iωt*) used by COMSOL is adopted like in [1].

1.4 Outline of the theory and related key issues

QNMPole is particularly useful for designing micro or nanoresonators, or optical nanoantennas, since analytical solutions and simulation provide great insights into how these devices operate. Since nanoresonators are essentially made of metal and can have different shapes, their simulation should rely on a software that can represent their geometry and model their electromagnetic properties accurately. **QNMPole** relies on COMSOL Multiphysics®, its RF Module, and MATLAB®.

Classically, to solve Maxwell's equations, one uses a particular excitation of the resonators, i.e. a given incidence, wavelength, and polarization of a light beam impinging on the resonator. However, the whole numerical simulation has to be redone each time the excitation field changes. Then the numerical load may be heavy, and, above all, the computed results obtained with brute-force calculations may still hide a great deal of knowledge about the physical mechanisms at play.

Using the striking of a tuning fork as an analogy for light excitation of a nanoresonator, it is possible to understand that any stroke will more or less excite the same vibration modes of a fork. The modes represent an intrinsic characteristic of the resonator, which do not depend on the excitation. If one is able to find these modes (they are called quasi-normal modes) and understand how they are excited, then it is possible to describe the interactions between the resonator and its environment much more easily and intuitively, and without the need to rely on brute-force calculations.

The approach adopted by **QNMPole** is exactly this one. It permits the normalization of the modes and allow to compute their excitation coefficients, simply by evaluating a volume integral [1]. This part is crucial as it results in a rapid and analytical method to calculate the electromagnetic field scattered by the resonator, and all the associated physical quantities, such as the scattering and absorption cross sections and the radiation diagram depicted in figure 3.

In more mathematical terms, we consider a field $\mathbf{E}_b(\mathbf{r},\omega)$ that is incident on the nanoresonator. $\mathbf{E}_b(\mathbf{r},\omega)$ can be a plane wave (to calculate cross-sections) or the field radiated by a dipole source (to calculate the LDOS or the Purcell factor). The optical response of the resonator (e.g. the scattered field) $\mathbf{E}_S(\mathbf{r},\omega,e)$ can be written with a modal expansion of the form

$$\mathbf{E}_S(\mathbf{r},\omega,\mathbf{E}_b) = \sum_m \alpha_m(\omega,\mathbf{E}_b)\tilde{\mathbf{E}}_m(\mathbf{r},\tilde{\omega}_m), \quad (1)$$

where $\tilde{\mathbf{E}}_m$ denotes the electric-field map of the normalized QNM *m*, $\tilde{\omega}_m$ is the mode complex frequency, $2Q = \mathrm{Re}(\tilde{\omega}_m)/\mathrm{Im}(\tilde{\omega}_m)$, and the α_m's are the excitation coefficients that **analytically** depend on the incident field. This implies that, once the resonant modes of a nanostructure are calculated, the optical response is known **analytically** (i.e. by numerical computation of simple overlap integrals between the incident field $\mathbf{E}_b(\mathbf{r},\omega)$ and the QNM field $\tilde{\mathbf{E}}_m$ [1]) for any instance of the excitation field and the physical understanding is immediate and unambiguous since the mode expansion explicitly depends on the

excitation parameters.

To summarize, the QNM computation and normalization (the important part) are performed with **Script_QNM_web.m**, a Matlab script that requires only a single input parameter, the initial guess of the complex frequency. The script automatically performs a pole search in the complex frequency plane. It is ***fully-documented*** and provides a ***pedagogical introduction*** of how to calculate of QNMs with Matlab-COMSOL livelink. **Script_QNM_web.m** uses two other short Matlab functions **omega_generation.m** and **setCOMSOL_ComplexFreq.m**.

Script_cross_sections_QNM.m is a Matlab script that uses the normalized QNM field obtained with **Script_QNM_web.m** to calculate the absorption and extinction cross sections of nanoparticles [1]

$$\sigma_A(\omega,\mathbf{E}_b) = -\frac{\omega}{2S_0}\iiint_{V_{res}} \mathrm{Im}\big(\varepsilon_\infty(\mathbf{r})\big)|\mathbf{E}_S(\mathbf{r},\omega)+\mathbf{E}_b(\mathbf{r},\omega)|^2 d^3\mathbf{r}\,, \tag{2}$$

$$\sigma_E(\omega,\mathbf{E}_b) = -\frac{\omega}{2S_0}\iiint_{V_{res}} \mathrm{Im}\big\{(\varepsilon_\infty(\mathbf{r})-\varepsilon_\mathrm{b})\big(\mathbf{E}_S(\mathbf{r},\omega)+\mathbf{E}_b(\mathbf{r},\omega)\big)\cdot\mathbf{E}_b^*(\mathbf{r},\omega)\big\} d^3\mathbf{r}\,. \tag{3}$$

The computation relies on analytical formulas of the α_m [1]

$$\alpha_m\left(\omega,\mathbf{E}_b\right) = \frac{-\omega\iiint \Delta\boldsymbol{\varepsilon}(\mathbf{r},\omega)\mathbf{E}_b(\mathbf{r},\omega)\cdot\tilde{\mathbf{E}}_m(\mathbf{r})\,\mathrm{d}^3\mathbf{r}}{(\omega-\tilde{\omega}_m)}\,. \tag{4}$$

Essentially, **Script_cross_sections_QNM.m** calculates the three volume integrals of Eqs. (2)-(4), in which $\Delta\boldsymbol{\varepsilon}(\mathbf{r},\omega)$ is the permittivity contrast $\Delta\boldsymbol{\varepsilon}(\mathbf{r},\omega)=\boldsymbol{\varepsilon}(\mathbf{r},\omega)-\boldsymbol{\varepsilon}_b$ of the nanoparticle, with $\boldsymbol{\varepsilon}_b$ the permittivity of the background medium surrounding the particle.

1.5 Provided programs

File name	Succinct description	Documented-didactical script? (Y/N)
Script_QNM_web.m	Main Matlab script for calculating and normalizing the QNMs. The script is fully-documented (including the Matlab-COMSOL livelink functions) and pedagogical.	Y
omega_generation.m	Matlab function required to perform the pole search in the complex frequency plane. It generates a new complex frequency triplet from a triplet of complex frequencies. With the Appendix 2 in [1], the script is fully understandable.	Y
setCOMSOL_ComplexFreq.m	Short Matlab script to artificially allows COMSOL to perform calculations at complex frequencies, using a trick which consists in modifying the permittivity and permeability distributions, see Appendix 1 in [1].	N
Script_cross_sections_QNM.m	Matlab script that uses the normalized QNM field calculated with **Script_QNM_web.m** to calculate the scattering and absorption cross sections of nanoparticles. The computation	Y

	relies on analytical formulas, see [1].	
Script_cross_sections_FV.m	Matlab script to calculate the scattering and absorption cross sections of nanoparticles with full-vectorial (FV) calculations using COMSOL. No QNM is involved in this script.	Y partly
nanorod_OE2013_V5dot2.mph	COMSOL model sheet used by **Script_QNM_web.m** and **Script_cross_sections_QNM.m**. **V5dot2** hold for the COMSOL Version 5.2	_
nanorod_OE2013_full_V5dot2.mph	COMSOL model sheet used by **Script_cross_sections_FV.m. Full** means thatno symmetry is used.	_

2. TUTORIAL: COMPUTATION AND NORMALIZATION OF QNM FIELD

We recommend that the user starts with the nanorod example that is provided to become familiar with **QNMPole** before calculating a QNM for its own nanoresonator. To calculate and normalize QNMs, follow the following steps:

1/ Build on a COMSOL model sheet for your problem (see details below), or first use the supplied nanorod model sheet **nanorod_OE2013_V5dot2.mph** (a single COMSOL versions exist in 5.2).

2/ Open the Matlab script **Script_QNM_web**, and make sure that the Matlab functions **setCOMSOL_ComplexFreq.m** (that adapts the model sheet to make computations at complex frequencies) and **omega_generation.m** (that generates the pole evaluation frequency triplet) are in your Matlab working path.

3/ In **Script_QNM_web**, check a few parameters in the ***Physical parameters, Numerical critical parameters, COMSOL parameters, Numerical parameters*** sections of the program. In particular, please check the following parameters

- lambda_guess, the initial guess value of the complex QNM wavelength
- delta: The initial frequency triplet is defined by (f0=2πc/lambda_guess, f1=f0*(1- delta), f2=f0*(1+ delta)). The convergence performance of the pole search depends on the choice of delta and lambda_guess.
- omega_var_name: name of the frequency parameter used in the tag "materials" in the COMSOL sheet, and that defines epsilon(omega) and mu(omega)) in rad/s.
- max_solv_iter: close to the pole, the matrix inverted by COMSOL to solve Maxwell's equations becomes singular. max_solv_iter provides a maximum number of iterations for the COMSOL solver and prevents COMSOL to run indefinitely for nothing when the complex frequency is very close to the pole. The value should not be too small since max_solv_iter is also used for the initial iterations at complex frequencies that are not closed to the pole.

4/ Run the program. For each iteration at a complex frequency, the program calculates the total and background fields, to obtain by difference the scattered field necessary to normalize the QNM.

5/ Computation Progression is displayed in Matlab command window, via COMSOL progress window. If one wants to interrupt the COMSOL calculation, for instance because the frequency is already very close to the pole (see the parameter max_solv_iter), we recommend to use the COMSOL progress window. In this

way, the Matlab program still continues, to further normalize the field and plot the final data, the iteration convergence plots of the estimated pole frequency and the normalized field, the divergence of the calculated field as a function of the iteration number, and the final normalized QNM-field in the xOz plane.

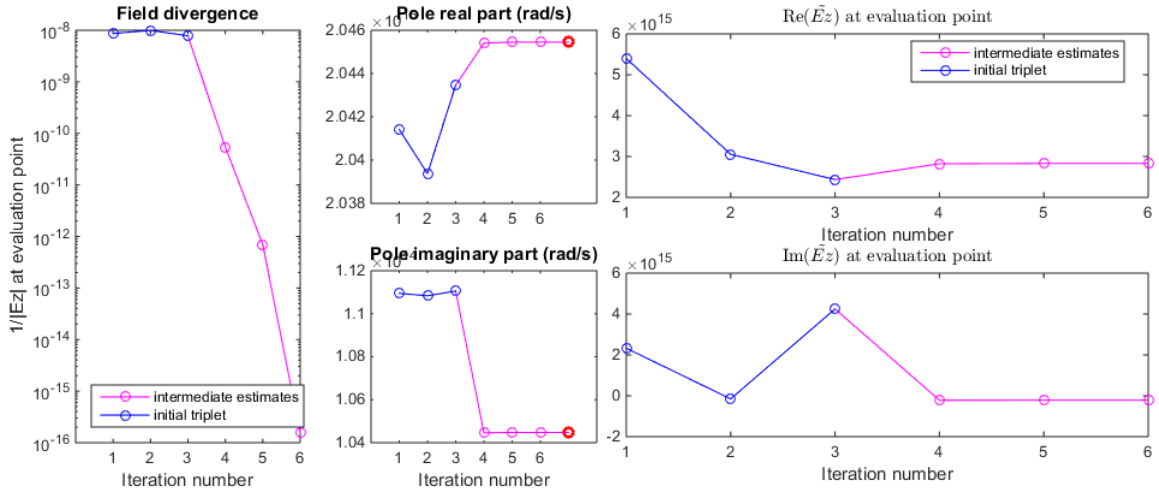


Figure 2. **Final plots provided by the program Script_QNM_web**. (left): Inverse of the *scattered field* computed at every iteration at the evaluation point; the inverse value should decrease by several decades to guaranty convergence as one approaches the pole. (middle): Convergence of $Re(\tilde{\omega}_m)$ and $Im(\tilde{\omega}_m)$ as one approaches the pole. A plateau should be observed. (right): Convergence of the real and imaginary parts of the *normalized QNM electric field* (z-component) at the evaluation point. A plateau should be observed. Note that the Purcell factor at the evalutation point is directly linked to the value of the field there [1]; thus the convergence performance also indicates the accuracy expected for the Purcell factor.

3. TUTORIAL: CALCULATE CROSS-SECTIONS USING QNM DATA

To calculate the scattering and extinction cross-sections of the nanoresonator with the QNM, we provide a Matlab script example, **Script_cross_sections_QNM.m**. The script is documented; it calculates the scattering and absorption cross-sections for a z-polarized incident plane wave, propagating along the x-direction. It is valid only for resonator that are composed of a single material ($\Delta\boldsymbol{\varepsilon}(\mathbf{r},\omega)$ should be inderendent of $\mathbf{r}$ in Eqs. 2-4), but it can be easily modified to deal with more general cases for which the resonators are composed of two or more materials.

For the calculation, follow the following steps:

1/ Open the Matlab script **Script_cross_sections_QNM.m**. This program can be used only if the QNM computation has been previously calculated and normalized using **ScriptQNM_web.m** , to respect some specific data format.

2/ Check the ***Numerical parameters*** in the Matlab script, especially the name of the material composing the resonator (tag in COMSOL model sheet), and the wavelengths for which the cross-sections are going to

be calculated. Beware of symmetries that are documented in the script.

3/ Run the program; it first initializes COMSOL to calculate volume integrals and related quantities, then calculates the overlap integrals (see Eqs. 2and 3) with a loop on the wavelength, and finally calculates the cross-sections and plot them.

4. TUTORIAL: CALCULATE CROSS-SECTIONS USING FULLY-VECTORIAL CALCULATIONS

The Matlab script **Script_cross_sections_FV.m**, which can be used independently of all other programs, calculates the scattering and absorption cross-sections for a z-polarized plane wave, propagating along x-axis. The script uses the COMSOL model sheet **nanorod_OE2013_full_V5dot2.mph** (a single COMSOL versions exist in 5.2). In the model, we define a closed surface surrounding the resonator (not overlapping the PMLs), with a mesh dense enough to accurately calculate the Poynting vector flux (then the scattering cross-section). We advise to define this surface as a sphere, and the geometric selection attached to the surface as an "explicit selection " of all the boundaries of this sphere.

The "scattered field formulation" with an incident palne-wave is used by the model sheet. The user cannot exploit any symmetries of his/her geometry with the Matlab script **Script_cross_sections_FV.m**, unless he/she modifies the code for this purpose. The script can only be used if the resonator is made of one sole material, but it can easily be modified to deal with resonators made of several materials.

For the calculation, follow the following steps:

1/ Open the Matlab script **Script_cross_sections_FV.m**.

2/ Check the ***Numerical parameters*** and the ***COMSOL parameters*** in the Matlab script, especially the integration surface selection used to calculate the scattering cross-sections.

3/ Run the program; it first initializes COMSOL, then runs field simulations and calculates cross-sections in loop on the wavelength, and finally plots the results.

5. FREQUENTLY ASKED QUESTIONS

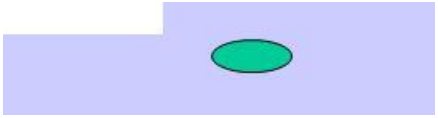
How to know that the complex resonant value is accurately calculated?

The program **Script_QNM_web.m** plots the inverse of the QNM-field computed at each iteration at some evaluation point. This field should diverge as $\left(\omega-\tilde{\omega}_m\right)^{-1}$ so that if the plot of the inverse decreases of *n* order or magnitude, one may estimate that $\tilde{\omega}_m$ is computed with a *n*-digit accuracy. In general, we consider that when a drop of 3-5 decades of the inverse of the QNM-field is realized, good accuracy is achieved.

How to check that the QNM is well normalized?

The program **Script_QNM_web.m** plots the normalized QNM-field computed at each iteration at some evaluation point. The real and imaginary part of the normalized QNM-field should exhibit a plateau as the iteration loop increases.

Can we calculate the QNM of nanoparticles that are embedded on a surface or a complex structure?
Consider the following case, in which a green scatterer is embedded in a step like substrate.



We can use **QNMPole** to calculate the QNM resonance mode. At each iteration, one should calculate the total field radiated by a source and the background field obtained with the same source by removing the particle. We will obtain a resonance frequency that will be different from that of the isolated nanoparticle.