Supporting Information

Modeling Mesoporous Nanoparticulated TiO_2 Films through Nanopolyhedra Random Packing

Loriano Storchi,[a*] *Filippo De Angelis,* [b*] *Francesca Nunzi*[b,c]

[a] Dipartimento di Farmacia, Universita' G. d'Annunzio, via dei Vestini
31, 66100 Chieti, Italy.

[b] Computational Laboratory of Hybrid/Organic Photovoltaics (CLHYO), Istituto CNR di Scienze e
Tecnologie Molecolari, via Elce di Sotto 8, I-06123 Perugia, Italy.

[c] Dipartimento di Chimica, Biologia e Biotecnologie, Università di Perugia, via Elce di Sotto 8, I-
06123 Perugia, Italy.

*Corresponding author. E-mail: loriano@storchi.org, filippo@thch.unipg.it

Tel.: + 39 75 585 5523

Computation Details

Here we report and quote the frameworks employed to develop the whole computational approach envisaged for the computer simulation of the mesoporous nanoparticulated TiO_2 film. An overview of the entire work-flow is also summarized in Scheme 1. Python[1] has been adopted as main programming language, being a natural choice in term of code readability and reusability.[2] Afterwards, Numpy[3, 4] is the Python package used for the numerical application. A fundamental aspect of the developed computer simulation has been the ability to visualize step by step the main aspects of the procedure by means of the VTK (The Visualization Toolkit) software.[5] VTK is an open-source, freely available software system for 3D computer graphics, image processing and visualization, that includes a Python interpreted interface layer we used in our work. It is worth nothing that all the figures reported in the following sections have been realized using the Python API of the VTK package. The whole numerical procedure we implemented, including all the visualization tools, is made of more than ten thousands Python code lines. Interestingly, the implemented tools represent an optimal reference for any future optimization and evolution of the procedure.

One of the most demanding part of the procedure, both from a computational and implementation time point of view, is the estimation of the pore size distribution (PSD) function, that is a fundamental parameter characterizing the dense particle packing. Generally, the PSD function is computed by simply searching the biggest sphere that can be centered in a point within the void, without overlapping the solid phase. In the present work we implemented two different approaches to provide a better description of the pores size and shape distribution: the triangulation (TR)[6, 7] and the Monte Carlo (MC) integration methods[8, 9] (see Figure 1). The TR method is a two-dimensional approach where we randomly place a number of points in the pore space. Then, using the point as a common vertex of the further triangles, we basically subdivide the pore surface into a number of triangles. This provides a simple way to estimate the cross section of the pore, and thus a quick approach to estimate the PSD function. The adopted MC method is that reported by

Bhattachary and Gubbins.[8] For any given point $P(x,y,z)$ in the pore space we look for the largest sphere that can be placed in the pore, without overlapping the solid phase, and containing P. This is a nonlinear optimization problem that can be solved using any appropriate nonlinear programming (NLP) routine, like SOLVOPT.[10] Specifically, because all the code we developed has been written in Python, we used pyOpt,[11] a Python-based package suitable for formulating and solving nonlinear constrained optimization problems. PyOpt offers a wide variety of NLP routines accessible through a common interface, we finally used the cited SOLVOPT, as suggested by Bhattachary and Gubbins.[8]

References

1. Rossum, G. V. *Python reference manual*. PythonLabs: Virginia, USA, 2001.
2. Merali, Z. Computational science: ... Error. *Nature* **2010,** *467,* 775-777.
3. Dubois, P. F.; Hinsen, K.; Hugunin, J. Numerical python. *Comput. Phys.* **1996,** *10,* 262-267.
4. Oliphant, T. E. *Guide to numpy*. Brigham Young University, 2006.
5. Schroeder, W.; Martin, K.; Lorensen, B.; Hall, P. *The visualisation toolkit*. Kitware: 2002; p 1-930934.
6. Primera, J.; Hasmy, A.; Woignier, T. Numerical study of pore sizes distribution in gels. *J. Sol-Gel Sci. Technol.* **2003,** *26,* 671-675.
7. Primera, J.; Woignier, T.; Hasmy, A. Pore structure simulation of gels with a binary monomer size distribution. *J. Sol-Gel Sci. Technol.* **2005,** *34,* 273-280.
8. Bhattacharya, S.; Gubbins, K. E. Fast method for computing pore size distributions of model materials. *Langmuir* **2006,** *22,* 7726-7731.
9. Gelb, L. D.; Gubbins, K. E. Pore size distributions in porous glasses: A computer simulation study. *Langmuir* **1999,** *15,* 305-308.
10. Kappel, F.; Kuntsevich, A. V. An implementation of shor's r-algorithm. *Comput. Optim. Appl.* **2000,** *15,* 193-205.
11. Perez, R. E.; Jansen, P. W.; Martins, J. Pyopt: A python-based object-oriented framework for nonlinear constrained optimization. *Struct. Multidiscip. O.* **2012,** *45,* 101-118.

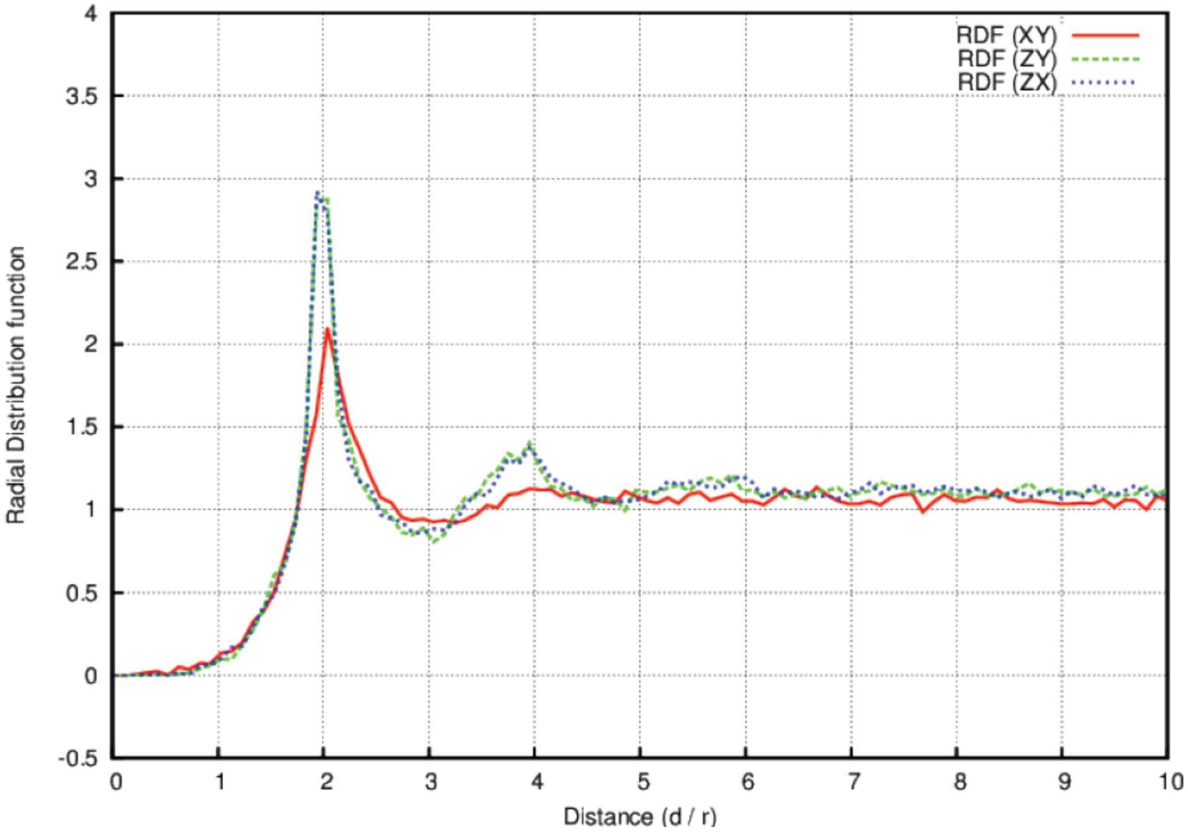

Figure S1. Computed 2D Radial Distribution Function along the three planes XY, ZX and ZY.