- Start with your core question: “What domain size delivers a stable effective diffusivity (REV)?” Structure around Problem → Approach → Evidence → Pending Work.
- **Problem**: one concise slide on why REV matters for transport simulations, the density targets (1e12, 1e13), and the stability criterion you’ll enforce (e.g., ΔD_eff/L convergence and narrow CIs).
- **Approach**: diagram the pipeline—Voronoi generator fixes, Dirichlet/Neumann solves, resume-capable sweep driver, and data QC steps. Include a simple flow graphic instead of raw code.
- **Evidence**: hold off until the sweeps finish; today’s plots look noisy because most L bins still sit at 40 samples. Once each L hits ≥100 runs, rebuild figures using per-L mean ±95 % CI or relative SEM thresholds.
- **REV Determination**: dedicate a slide to the criterion. Recommended: declare REV when both boundary-condition means change <5 % between successive L points and SEM/mean <2 %. Use a table that shows metrics vs. L for both densities, highlight the first L that satisfies all checks.
- **Status & Next Steps**: summarize remaining deficits (1e12 needs 1 360 runs, 1e13 needs 1 729), outline the queued batches, then state the final analysis tasks (aggregation notebook, report-ready plots, REV call).

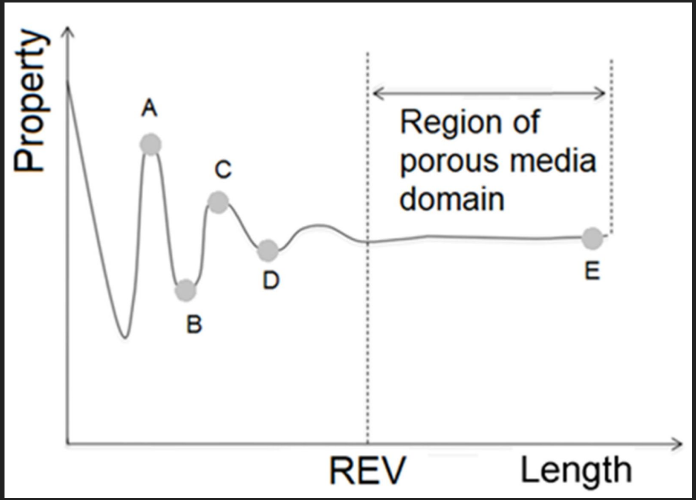
The Impact of Domain Size on Effective Diffusion in a 3D Voronoi Network

Arthur Amalir
MULTISCALE COMP SCI & ENGIN
Fall 2025

Problem

For simple diffusion problem in generic porous media,

What **domain size** delivers a stable effective diffusivity (**REV**)?



Source: Gaspareto et al., 2023 (https://www.mdpi.com/2077-0472/13/3/736)

Introduction

Objective

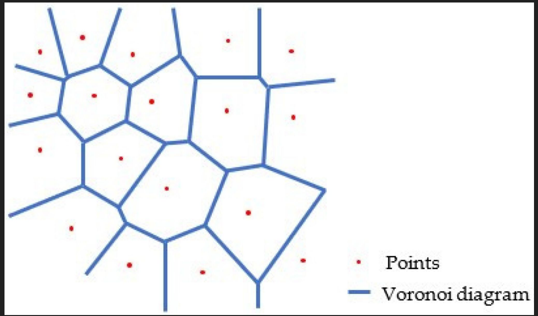
Determining the scale (and existence) of a REV

I.e the smallest domain where bulk transport properties stop changing with size → cheaper multiscale simulations.

Method

1. Generate 3D porous media (Voronoi networks)
2. Compute steady-state Fickian diffusion (using OpenPNM.)
3. Track convergence of transport parameters as domain size increases.

Outcome:
→ **A validated REV curve** & a dataset enabling ML prediction of diffusivity from network descriptors.



Voronoi tessellation
(Source: Lu et al., 2022)

Pipeline

1. **<u>Network generation:</u>** Simple Fickian diffusion of Air on a given network (no chemical interactions → only base diffusivity and gradients) with
    i. Nuemann BCs
    ii. Dirichlet BCs
2.

Source: Gaspareto et al., 2023 (https://www.mdpi.com/2077-0472/13/3/736)

I. Network Generation

1.

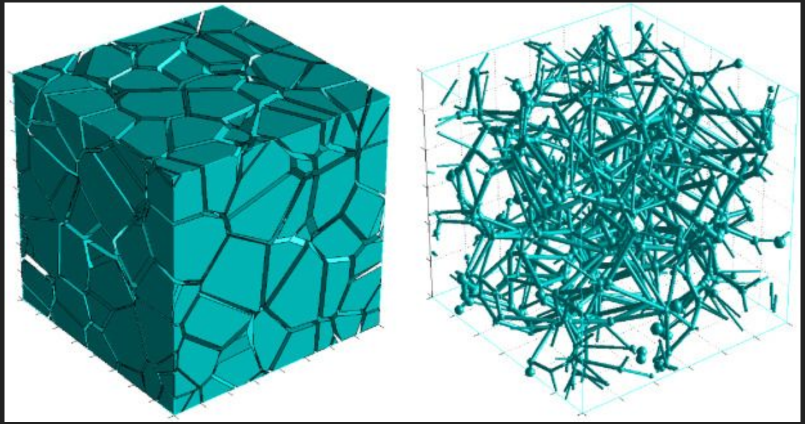
3D Porous Networks

Objective:

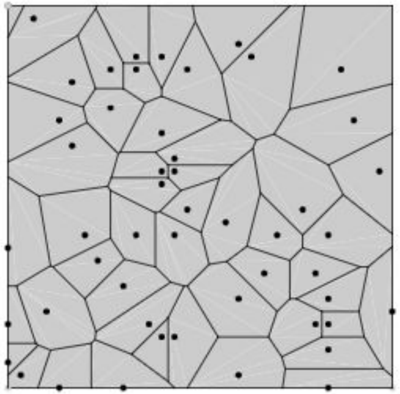
Generate a statistically realistic pore network (suitable for diffusion simulations + w/ controlled geometry.)

How:

Voronoi 3D porous networks of a **constant density and equal pore size distribution** → hypothetical material consisting of solid "tubes" increasing in scale



Granular and tubular geometries of 3D pore networks w/ tube size distributions. (Xiao & Lin, 2015)



The partitioning of a plane with n points into convex polygons such that each polygon contains exactly one generating point and every point in a given polygon is closer to its generating point than to any other. A Voronoi diagram is sometimes also known as a Dirichlet tessellation. The cells are called Dirichlet regions, Thiessen polytopes, or Voronoi polygons.

Formal definition of Voronoi network (source: https://mathworld.wolfram.com)

3D Voronoi Network Generation

1. Seed-point generation

- **Domain size: MAIN VARIABLE**
- **Pore density** → number of points (N = rho * V) (fail-fast if (N<10)) (CONSTANT FOR A GIVEN SWEEP)
- **Random uniform distribution of seed points**

2. Voronoi tessellation: standard method for representing disordered porous materials (Li et al. 2012)

- Voronoi cells → pores
- Shared faces → throats

3. Boundary identification + boundary pores (tol scaled with domain size)

4. Stochastic pore & throat diameters

- **Pores:** lognormal ((s=0.25), scale (20 micrometers)): MCLT
- **Throats:** Weibull ((c=2.5), scale (8 micrometers): EVT

5. Physical constraints + model regeneration

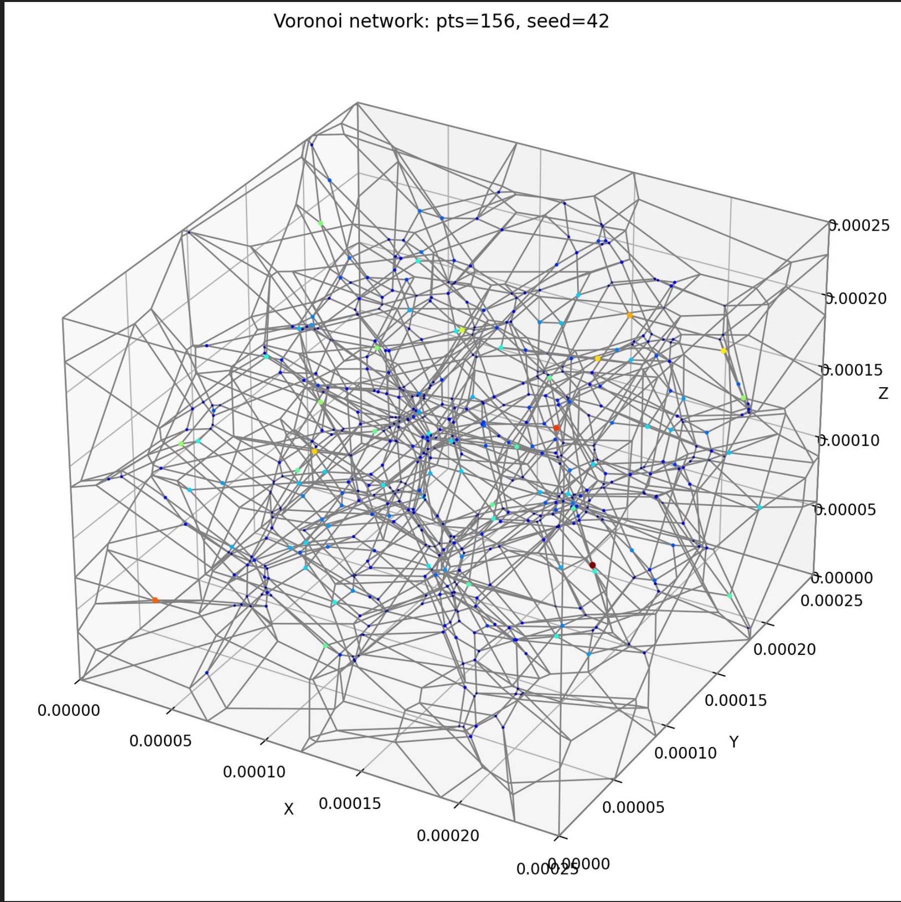
- Enforce (d_throat <= L_throat) + other checks
- Regenerate all geometry

Density Function	$f(x) = \dfrac{\alpha}{\beta^{\alpha}} x^{\alpha-1} e^{-\left(\frac{x}{\beta}\right)^{\alpha}}$ $x \geq 0, \quad \alpha > 0, \quad \beta > 0$
Cumulative Distribution Function	$P(X < c) = \int f(x)\ dx = 1 - e^{-\left(\frac{c}{\beta}\right)^{\alpha}}$ $P(X > c) = \int f(x)\ dx = e^{-\left(\frac{c}{\beta}\right)^{\alpha}}$
Mean	$\mu = E(X) = \beta \cdot \Gamma\left(1 + \dfrac{1}{\alpha}\right)$
Variance	$\sigma^2 = V(X) = \beta^2 \left\{ \Gamma\left(1 + \dfrac{2}{\alpha}\right) - \left[\Gamma\left(1 + \dfrac{1}{\alpha}\right) \right]^2 \right\}$

Weibull distribution def. (source: https://calcworkshop.com)

Why the log-normal distribution is usually the better model for original data. As discussed above, the connection between additive effects and the normal distribution parallels that of multiplicative effects and the log-normal distribution. Kapteyn (1903) noted long ago that if data from one-dimensional measurements in nature fit the normal distribution, two- and three-dimensional results such as surfaces and volumes cannot be symmetric. A number of effects that point to the log-normal distribution as an appropriate model have been described in various papers (e.g., Aitchison and Brown 1957, Koch 1966, 1969, Crow and Shimizu 1988). Interestingly, even in biological systematics, which is the science of classification, the number of, say, species per family was expected to fit log-normality (Koch 1966).

Extract from Limpert et al., 2001



Example of generated 3D network (seed = 42, density = 1e13, domain size 250e-6m)

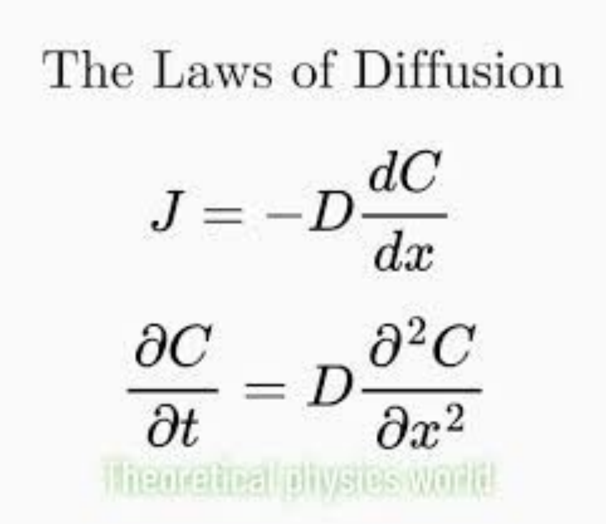
II. Diffusion Solvers

1. OpenPNM solver {openPNM.FickianDiffusion()}
2. Dirichlet & Neumann Solvers

OpenPNM.**FickianDiffusion()**

Purpose:

- Linear solver simulating binary diffusion in porous networks
    - Accounts for porosity and tortuosity (geometric constraints) → Diff. is a *f* of empty space
    - Computes **effective diffusivity via Fick's law:**



Fick's laws of diffusion
(Source: physicsworld.com)

Notes:

Implicit so unconditionally stable.

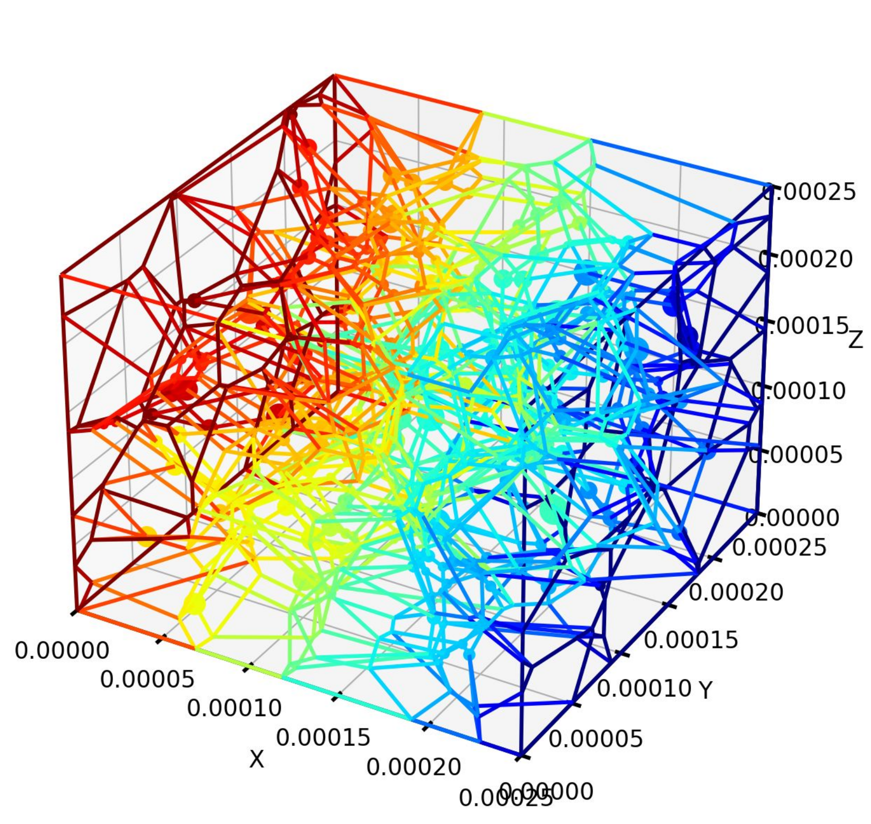
Dirichlet & Neumann Solvers

Objective

- **Predict REV while negating boundary effects**
    - **PBCs:** Unpredictable, wrapping yields non physical results
    - “Buffer zones”: computationally intensive, effectively the same as increasing domain size

So,

- **2 complementary BC regimes to bracket the diffusivity of the Voronoi medium.**
    - Dirichlet and Neumann represent the two cheap and extremal ways to drive transport in a finite domain. Provides checks, bias detection, and bounds on REV behavior.



Example of a diffusion simulation using Dirichlet BCs (seed = 42, density = 1e13, domain size 250e-6m; Cin = 1, Cout= 0)

1.Dirichlet BC (Concentration-Driven)

- Apply fixed concentrations: Cin=1,Cout=0

→ the solver generates whatever flux geometry permits.

- Compute: $$D_{\text{eff}}^{(D)} = \frac{Q\,L}{A\,\Delta C}$$

Justification (Upper Bound Tendency)
Dirichlet tends to overestimate Deff:

- It forces a *linear macroscopic gradient*, even if the microscopic structure would create bottlenecks.

2. Neumann BC (Flux-Driven)

- Impose total flux Qin at inlet.
- Fix only one value at outlet Cout=0 to remove null-space. (WTF IS THIS)

→ Result: the solver produces whatever concentration gradient is necessary to sustain the imposed flux.

- Compute: $$D_{\text{eff}}^{(N)} = \frac{Q_{\text{in}}\,L}{A\,(C_{\text{in,avg}} - C_{\text{out}})}$$

Justification (Lower Bound Tendency)
Neumann tends to underestimate Deff

- It allows strong local concentration drops in narrow throats or poorly connected clusters.
- The system self-adjusts with a larger gradient where the geometry is restrictive.

3. Why Both?

Using both Dirichlet and Neumann BCs provides:

- **Upper and lower bounds:**

$$D_{\text{eff}}^{(N)} \leq D_{\text{eff}}^{(\text{periodic})} \leq D_{\text{eff}}^{(D)}$$

- **Detection of geometric degeneracies** (large gap between the two = poor REV).
- **Quality control**: if both converge as domain size increases, the REV is reached.
- **Benchmarking**: periodic BCs (true bulk behavior) must lie between them.

4. Implementation Consistency

Both scripts share:

- Same repaired Voronoi geometry.
- Same explicit diffusive conductance:

$$g = DA/L.$$

- Same detection of conductive cluster to remove dead-end artifacts.
- Same solver backend (PyPardiso) for comparability.

Thus the pair acts as a consistency check, not two unrelated physics problems.

Sweep Driver

• Loop over domain sizes and seeds.

• For each configuration:

– build network

– compute geometry sanity fixes

– run Dirichlet

– run Neumann

– store

(

D

e

f

f

D

,

D

e

f

f

N