Deposition of flame-made nanoparticles on porous media

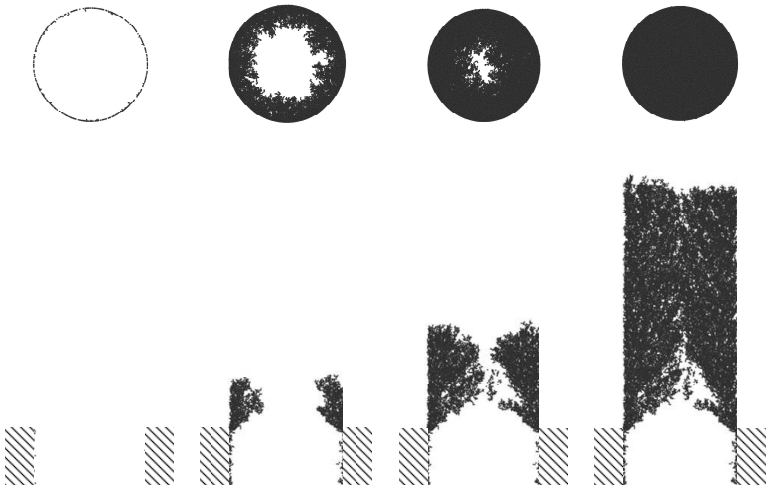
Elmøe, Tobias Dokkedal; Grunwaldt, Jan-Dierk; Dam-Johansen, Kim; Pratsinis, Sotiris E.

Publication date:
2008

Document Version
Publisher's PDF, also known as Version of record

Deposition of flame-made nanoparticles on porous media

PhD thesis



Author: Tobias Dokkedal Elmøe
Supervisors: Jan-Dierk Grunwaldt[1]
Kim Dam-Johansen[1]
Sotiris E. Pratsinis[2]

[1] Department of Chemical and Biochemical Engineering
Technical University of Denmark
[2] Particle Technology Laboratory
ETH Zürich, Switzerland

31st May 2008

Contents

Preface

This dissertation is a partial fulfillment of the requirements of the Ph.D. degree. The work was carried out at the former Aerosol Laboratory, Department of Chemical and Biochemical Engineering, Technical University of Denmark (DTU), as well as at the Particle Technology Laboratory, Department of Process Engineering, ETH Zürich, Switzerland. My supervisors were Prof. Dr. Jan-Dierk Grunwaldt (DTU), Prof. Dr. Kim Dam-Johansen (DTU) and Prof. Dr. Sotiris E. Pratsinis (ETH Zürich).

I would like to thank all my supervisors for all their dedication to the project and in particular Prof. Pratsinis for his invaluable guidance, that I have received throughout my stay at ETH. I am very happy, that even though there were several rearrangements during time of the project, I have ended up with what I regard as the best possible outcome in terms of qualified and inspirational guidance, with my current three supervisors.

I would also like to thank my past and present colleagues, in particular Johnny Johansen, that I had the pleasure of working with for the first year of my PhD, and also Antonio Tricoli and Björn Schimmöller without whose friendship and support I would have spend several hours more pulling out hairs of my head.

Furthermore, I thank the entire PTL team for not only making my stay at Zürich a very productive one, but also for making me feel welcome socially as one of their own.

I would like to thank my former supervisors Tue Johannessen and Hans Livbjerg and the Danish Council of Technogical Research (STVF) for financial support.

Finally, I also would like to thank my girlfriend Milena, for helping me read proof and for always cheering me up in the stressed hours.

Tobias Dokkedal Elmøe
Zürich, Switzerland
May 2008

Summary

Nanoparticles in porous ceramic materials have several applications such as gas-sensors, catalysts, fuel cell anodes and diesel soot traps. One rapid technique for the production of these is the *direct* deposition technique, where particles are directly deposited onto a substrate and a particle cake is formed. In this work, the direct deposition of flame-made nanoparticles by filtration was studied. Emphasis was put on the characterization and stabilization of the particle cakes, as any practical application will be directly related to the final morphology and mechanical stability of the these. Furthermore, the understanding of how to influence the morphology during deposition by filtration, was studied in a novel modelling approach by Langevin dynamics (LD).

The cakes were deposited on porous α-alumina substrates and consisted of flame-made Al_2O_3 agglomerated particles with an average agglomerate mobility size of 27.9 nm. They were highly porous ($> 94 - 98$ % porosity) and as a consequence had a high capillary size between 160 - 227 nm. In the experiments, the Peclet numbers (Pe, ratio of diffusive to convective transport) were very similar (0.3 - 0.5), and it was not possible to observe any change in the morphology of the cakes. However, the modelling study showed that the porosity could be decreased from 99% to 85% by increasing the Pe number above 10 during filtration.

In order to improve the mechanical strength of the cakes, they were sintered at 500 - 1100 °C. Thick cakes maintained their overall integrity up to and including 900 °C, while thinner cakes showed a substantially increased gas-permeability at 900 °C. At 700 °C micro-cracks were observed, the number of which increased at 900 °C. One reason may be, that the micro-cracks formed due to contraction and expansion of the cake lodged in the substrate capillary mouth during the heating/cooling cycle of sintering. At 1100 °C, however, the cakes completely disintegrated due to a free-shrinkage of the initially highly porous cake.

To test the mechanical stability, the cakes were subjected to cyclohexane vapours at high relative saturation pressures. As-deposited cakes disintegrated at high relative saturation due to condensation of cyclohexane in the cake, which pulled the cake particles apart, as the capillary forces were orders of magnitude higher than the corresponding Van der Waals forces.

After sintering, the cakes displayed excellent stability toward high relative saturation and no degradation was observed. This was explained as a result of sintering necks between the cake constituent particles.

Finally, a novel modelling study on the deposition of monodisperse nanoparticles during filtration was carried out in the Pe range of 0.01 - 10. The porous substrate was modelled using a cylindrical capillary and deposition with the transition from capillary deposition to cake growth, was studied. The morphology was characterized quantitatively by the solid volume fraction (1-porosity) profiles, as well as qualitatively by visual inspection. At low Pe numbers, the cakes were fractal-like with a high porosity (99% at Pe = 0.01 for 50 nm particles), while at high Pe numbers, the porosity approached the ballistic limit (85% at Pe = 10 for 50 nm particles). The porosity of the initial deposit was different from the final porosity and after plugging of the substrate capillary had finished, a constant cake porosity was obtained. At low Pe, the lowest porosity was found to be at the inlet to the capillary mouth, while at higher Pe, the lowest porosity was reached in the filtration cake. For all Pe, plugging was seen to occur outside of the substrate capillary, which was in excellent agreement with experimental SEM observations. The plugging time (5 min) and cake porosity (95 %) for deposition of 25 nm particles agreed well with experimental findings (1.7 min and 94 - 97 % respectively) for deposition under similar conditions.

Based on the work, the possible novel application of the highly porous cakes in diesel soot filtration was discussed. The presented model may be used to optimize parameters in the filtration of soot, so as to obtain as high a degree of contact between the soot- and cake-particles as possible. Another novel application of the filtration method may be the direct coating in the capillaries of a porous substrate with nanoparticles. In this connection, the model may also be used to find optimal process parameters such as concentration, Pe number and choice of substrate capillary size.

Resumé på dansk

Nanopartikler i porøse keramiske materialer har en lang række anvendelsesmuligheder, som f.eks. gas-sensorer, katalysatorer, brændselscelle anoder samt diesel sod filtre. En hurtig metode til dannelsen af disse er ved *direkte* deponering, hvor partiklerne afsættes direkte på en substratflade og en partikelkage opbygges. I dette arbejde er direkte deponering ved filtrering af flammedannede partikler blevet studeret. Særlig vægt er lagt på karakteriseringen og stabiliseringen af partikelkagerne, eftersom enhver praktisk anvendelse af disse er direkte forbundet til morfologien samt den mekaniske stabilitet. Yderligere er forståelsen af, hvordan morfologien kan påvirkes under filtreringsprocessen blevet studeret i en ny modelleringstilgang ved anvendelsen af Langevin dynamik (LD).

Partikelkagerne blev deponeret på porøse α-alumina substrater og bestod af flammedannede agglomerede Al_2O_3 partikler med en gennemsnitlig agglomeratstørrelse på 27.9 nm. De var højporøse ($> 94-98$ % porøsitet) og havde grundet deraf en høj kapillærstørrelse mellem 160 - 227 nm. I eksperimenterne var Peclet tallene (Pe, forholdet mellem konvektiv og diffusiv transport) temmelig ens (0.3 - 0.5) og det var ikke muligt at konstatere nogen ændring i morfologien af partikelkagerne. Dog viste modelleringsstudiet, at porøsiteten kunne nedsættes fra omtrent 99 % til 85 % ved at øge Pe tallet til mere end 10 under filtreringen.

For at øge den mekaniske styrke af partikelkagerne blev de sintret ved 500 - 1100 °C. Tykke partikelkager beholdt deres overordnede helhed til og med 900 °C, mens tynde partikelkager udviste en væsentlig øget gaspermeabilitet ved 900 °C. Ved 700 °C blev der konstateret mikrosprækker hvis antal øgedes ved 900 °C. En mulig grund er, at mikrosprækkerne blev dannet under opvarming/afkølingsfasen i sintringsprocessen, som følge af sammentrækning og udvidelse af partikelkagen, der sad fast i kapillærmundingen. Ved 1100 °C blev de dog fuldstændigt ødelagte på grund af fri sammentrækning af de oprindeligt højporøse partikelkager.

For at afprøve den mekaniske stabilitet, blev partikelkagerne udsat for cyklohexan dampe ved et højt relativt mætningstryk. Netop deponerede partikelkager forvitrede ved et højt relativt mætningstryk, grundet kondensation af cyklohexan i partikelkagen, der trak kagepartiklerne fra hinanden, eftersom kapillærkræfterne var størrelsesordener højere end de tilsvarende

Van der Waalske kræfter. Efter sintring udviste partikelkagerne glimrende stabilitet imod høj relativ mætning og ingen forvitring blev observeret. Dette blev forklaret, som et resultat af sintringshalse imellem kagepartiklerne.

Endeligt blev deponeringen af monodisperse nanopartikler ved filtrering undersøgt i et et nyt modellingsstudium for en række Pe tal (0.01 - 10). Det porøse substrat blev modelleret som et cylindrisk kapillær og deponering med overgang fra kapillærdeponering til kagedannelse blev undersøgt. Morfologien blev karakteriseret kvantitativt ved faststofvolumenfraktionsprofilen, samt kvalitativt ved visuel iagtagelse. Ved lave Pe tal var partikelkagerne fraktallignende med en høj porøsitet (99 % ved Pe = 0.01 for 50 nm partikler), mens porøsiteten ved høje Pe tal nærmede sig den ballistiske grænseværdi (85 % ved Pe = 10 for 50 nm partikler). Porøsiteten af det førstdannede partikellag var forskelligt fra porøsiteten af det sidstdannede partikellag og efter tilstopning af substratkapillæret blev en konstant kageporøsitet opnået. Ved lave Pe tal blev den laveste porøsitet fundet ved kapillærmundingen mens den blev fundet at være i filtreringskagen for høje Pe tal. For samtlige Pe tal sås tilstopning at ske udenfor substratkapillæret, hvilket var i god overensstemmelse med eksperimentelle SEM observationer. Tilstopningstiden (5 min) samt kageporøsiteten (95 %) for deponering af 25 nm partikler var også i god overenstemmelse med eksperimentelle iagtagelser (hhv. 1.7 minut og 94 - 97 %).

På basis af dette arbejde er en mulig ny anvendelse af de højporøse partikelkager i diesel sod filtrering blevet diskuteret. Den fremlagte model kan benyttes til at optimere parametrene i filtrering af sod, således at der opnåes en så høj grad af kontakt imellem sod og kagepartiklerne som muligt. En anden ny anvendelsesmulighed af filtreringsmetoden er, at direkte belægge nanopartikler i kapillærerne på porøse substrater. I denne sammenhæng kan modellen anvendes til at bestemme optimale procesparametre, såsom koncentration, Pe tal samt valg af substrat kapillærstørrelse.

Nomenclature

α	Expansion coefficient (permeability) (unitless)
α_1	Expansion coefficient (permeability) before sintering (unitless)
α_2	Expansion coefficient (permeability) after sintering (unitless)
$\alpha_{\mathrm{TEC},1}$	Thermal expansion coefficient for material 1 (1/°C)
$\alpha_{\mathrm{TEC},2}$	Thermal expansion coefficient for material 2 (1/°C)
$\bar{d}_m$	Mass average agglomerate size (m)
$\bar{d}_n$	Number average agglomerate size (m)
β_c	Coefficient (2) for permeability measurements (cake) (m^3/Pa s)
β_s	Coefficient (2) for permeability measurements (substrate) (m^3/Pa s)
ΔP	Pressure drop during filtration (Pa)
ΔP_c	Pressure-drop across cake (Pa)
$\Delta P_{cake}(t)$	Pressure-drop through filter cake at time t (LD model) (Pa)
$\Delta P_{plug}(t_{plug})$	Pressure drop through plug at time t_{plug} (Pa)
$\delta_{\mathrm{capillary}}$	Length of substrate capillary (LD model) (m)
δ_{drop}	Drop domain height (m)
δ_{film}	Maximum thickness of particle film (LD model) (m)
δ_c	Thickness of cake (m)
δ_s	Thickness of substrate (m)
$\dot{n}$	Molar flow of gas (moles/s)
$\dot{N}_{cap}$	Flux of particles entering the capillary mouth (1/s)
$\dot{N}_{ins}$	Flux of particles entering the model domain (1/s)

ϵ_c	Cake porosity (unitless)
ϵ_s	Porosity of substrate (unitless)
$\epsilon_{c,1}$	Porosity of cake before sintering (unitless)
$\epsilon_{c,2}$	Porosity of cake after sintering (unitless)
$\epsilon_{c,avr}$	Average cake porosity (LD model) (unitless)
$\epsilon_{c,h}$	Cake porosity determined from cylindrical capillary model (unitless)
$\epsilon_{s,1}$	Porosity of substrate (method 1) (unitless)
$\epsilon_{s,2}$	Porosity of substrate (method 2) (unitless)
η_{agg}	Charge distribution for agglomerates (unitless)
η_{sphere}	Charge distribution for spheres (unitless)
λ_c	Coefficient (1) for permeability measurements (cake) (m^3/s)
λ_g	Gas mean free length of the (m)
λ_s	Coefficient (1) for permeability measurements (substrate) (m^3/s)
$\left(\frac{\epsilon}{\tau}\right)_c$	Porosity-tortuosity factor for cake (unitless)
$\left(\frac{\epsilon}{\tau}\right)_s$	Porosity-tortuosity factor for substrate (unitless)
$\mathbf{F}$	External force vector (N)
$\mathbf{R}$	Particle displacement vector (m)
$\mathbf{r}$	Particle position vector (m)
$\mathbf{u}$	Fluid velocity vector (m/s)
$\mathbf{V}$	Velocity displacement vector (m)
$\mathbf{v}$	Particle velocity vector (m/s)
$\mathbf{X}$	Random force vector (N)
$\mathbf{x}_0$	Start position vector of particle (m)
$\mathbf{x}_d$	Position vector of deposited particle (m)
$\mathbf{x}_e$	End position vector of particle (m)
μ_g	Gas viscosity (kg/ms)
ϕ_s	Solid volume fraction (unitless)

$\phi_{s,c}$ Constant solid volume fraction of cake (unitless)

ρ_0 Initial density (kg/m^3)

$\rho_{\text{substrate}}$ Density of solid in substrate (kg/m^3)

ρ_{water} Density of water (kg/m^3)

ρ_p Particle density (kg/m^3)

ρ_s Density of solid cake (kg/m^3)

σ Surface tension (N/m)

τ_f Characteristic time of coalescence (s)

A_c Flow area of cake (m^2)

A_s Logarithmic mean area of substrate (m^2)

B_{0c} d'Arcy permeability coefficient in the cake (m^2)

B_{0s} d'Arcy permeability coefficient in the substrate (m^2)

B_0 d'Arcy permeability (LD model) (m^2)

$C(d_m)$ Cunningham correction factor for agglomerate with mobility diameter d_m (m)

$C(d_p)$ Cunningham correction factor for particle of size d_p (unitless)

c^* Dimensionless drag force (unitless)

C_n Number concentration (m^{-3})

$D(d_p)$ Diffusion coefficient for particle of size d_p (m^2/s)

D_{agg} Diffusion coefficient of agglomerate (m^2/s)

$d_{\text{c,micro}}$ Micro-capillary size (m)

D_f Fractal dimension (unitless)

D_h Hydraulic diameter (m)

d_m Agglomerate mobility diameter (m)

d_p Particle or primary particle size (m)

d_{BET} Primary particle diameter determined from BET (m)

$d_{cc,1}$ Cake capillary size before sintering (m)

$d_{cc,2}$ Cake capillary size after sintering (m)

d_{cc} Cake average capillary size (m)

$d_{cr,2}$ Size of cracks (m)

d_{cs} Substrate capillary size (m)

D_{kc} Knudsen diffusion coefficient of gas in the cake (m^2/s)

D_{ks} Knudsen diffusion coefficient of gas in the substrate (m^2/s)

$d_{p,1}$ Primary particle diameter before sintering (m)

$d_{p,2}$ Primary particle diameter after sintering (m)

D_{qe} Charging equivalent diameter (m)

e Electronic unit charge (C)

E_1 Young modulus of material 1 (Pa)

E_2 Young modulus of material 2 (Pa)

$f(d_p)$ Friction factor for particle of size d_p (kg/s)

$F_{\text{capillary}}$ Capillary force (N)

f_s Fractional shrinkage (unitless)

h_{plug} Height of deposition at t_{plug} (m)

k_B Boltzmann constant (J/K)

K_E Conversion factor for charge distribution ($9 \cdot 10^9$ Nm^2/C^2)

K_s Constant group (permeability) (unitless)

Kn Knudsen number (unitless)

L Length of substrate (m)

L_c Characteristic length (m)

m Particle mass (kg)

$m_{\text{substrate}}$ Mass of substrate (kg)

m_{tot} Total aerosol mass-concentration (kg/m^3)

m_{water} Mass of water in substrate (kg)

M_g Molar mass of permeating gas (kg/mol)

N_p	Number of primary particles per agglomerate (unitless)
n_{agg}	Size-distribution (number) for agglomerates (#/m^3)
$n_{c,1}$	Number of cake capillaries before sintering (unitless)
$n_{c,2}$	Number of cake capillaries after sintering (unitless)
$N_{cr,2}$	Number of cracks per unit area after sintering (m^{-2})
$n_{cr,2}$	Number of cracks after sintering (unitless)
$N_{p,0}$	Number of primary particles per ideal agglomerate (unitless)
n_{sphere}	Size-distribution (number) for spheres (#/m^3)
P	Pressure (LD model) (Pa)
P_0	Pressure at low pressure side (permeability) (Pa)
P_1	Pressure at high pressure side (permeability) (Pa)
P_c	Partial pressure of cyclohexane (Pa)
P_E	Fraction of particles entering the capillary (unitless)
P_i	Cake-substrate interface pressure (Pa)
$P_{c,sat}$	Saturation pressure of cyclohexane (Pa)
Q	Compressible gas flow (LD model) (m^3/s)
Q_0	Flow-rate at pressure $P_0 = 10^5$ Pa (m^3/s)
$q_{c,1}$	Flow through cake capillary before sintering (m^3/s)
$q_{c,2}$	Flow through cake capillary after sintering (m^3/s)
$q_{cr,2}$	Flow through crack capillary after sintering (m^3/s)
$Q_{t,1}$	Total flow through substrate and cake before sintering (m^3/s)
$Q_{t,2}$	Total flow through substrate and cake after sintering (m^3/s)
R	Gas constant (J/mol K)
R_c	Capillary radius (LD model) (m)
R_g	Radius of gyration (m)
R_i	Inner radius of substrate (m)
r_k	Kelvin radius (m)

R_O Outer radius of substrate (m)

$r_{p,0}$ Primary particle radius (m)

S Relative saturation pressure of cyclohexane (unitless)

St Stokes number (unitless)

T Temperature (K)

t Parametric line value or time (unitless, or s)

T_{cell} Temperature of depositon cell (K)

t_c Parametric line value at point of particle-particle collision (unitless)

t_{plug} Plugging time (s)

t_{wc} Value of t at point of particle-wall collision (unitless)

U_0 Fluid/gas velocity (m/s)

v_p Particle volume (m^3)

$V_p(z)$ Volume of particles contained in slize at position z with thickness d_p (1 particle diameter), and radius R_c (m^3)

w_d Deposited mass (kg)

A Fractal prefactor (unitless)

$h^{(1)}_{\text{Quench}}$ Quench ring height above burner (m)

Pe Peclet number (unitless)

Q_{Air} Flow of air to burner (m^3/s)

Q_{Quench} Flow of quench air (m^3/s)

Q_{CH_4} Flow of methane to burner (m^3/s)

$Q_{N_2,\text{sat}}$ Flow of Nitrogen through saturator (m^3/s)

T_{sat} Temperature of saturator (K)

Chapter 1

Introduction

1.1 Particle production in flames

Production of particles in flames offers a fast and controllable way of synthesizing a variety of materials with unique properties, not available through traditional wet-phase techniques [1]. In particular, flame made particles are interesting for catalytic applications [2, 3], the most used industrial application, however, is found as paint pigments [4].

In general, three categories of flame units exists [3]: the VASF (Vapor Aerosol Flame Synthesis), the FASP (Flame Assisted Spray Pyrolysis) and the FSP (Flame Spray Pyrolysis).

There are several designs available, which have been studied extensively in the later years. Some of them have been listed below [5]:

- Diffusion flames (VASF): Fuel, oxidant (air or pure oxygen), and gaseous precursor are led separately to the flame front through the annuli of concentric tubes. Production rates of up to 200 g/h with a controlled particle morphology have been shown [6].
- Premixed flames (VASF): Fuel, oxidant and gaseous precursor are mixed before entering the flame. The premixed flame offers the possibility of producing mixed oxides with molecular scale mixing [7, 8].
- Spray flames (FSP): A liquid precursor is atomized through a nozzle and ignited by supporting flame-lets. Opposite the two previous flame devices, the main reaction enthalpy comes from the combustion of the precursor. Very high productions rates of single FSP units (1.1 kg/h) with controlled morphology have been shown [9].

1.2 Direct deposition of nanoparticles

Direct deposition of nanoparticles from an aerosol onto a substrate is a fast way of producing highly porous ceramic deposits for applications in fields

such as gas-sensors [10, 11], catalysts [12, 13], fuel cell anodes [14], membrane filters [15] and diesel soot traps [16], the latter of which is discussed in more detail below. The typical route of deposition is through thermophoresis, in which a cooled substrate is placed in the hot aerosol exhaust from a flame reactor, where the temperature difference causes particles from the gas to be deposited on the cold substrate. Another means of producing such porous deposits is by filtration through a porous substrate.

1.2.1 Diesel particulate filters

With increasingly stricter legislation on the particulate emission (soot) from diesel engines, the incitement has been given to develop particulate filters, which are cheap, effective and have a long operation life. The most used method in removing diesel particles is by a monolithic filter. By plugging every second channel, the exhaust gas is passed through the porous walls, as shown on figure 1.1 [17].

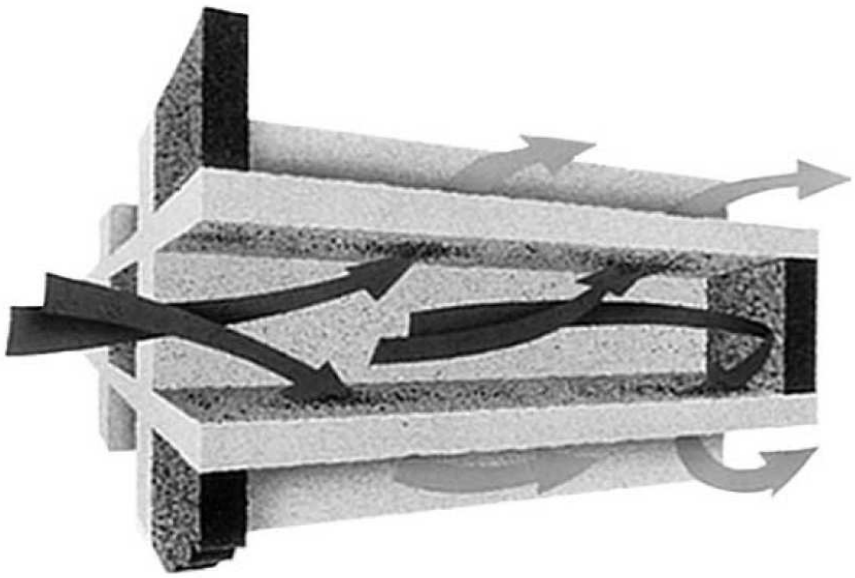


Figure 1.1 Principle of the diesel particulate filter with a "wall-flow" geometry. The exhaust gas is forced to pass through the wall of the filter, as every 2nd channel is sealed off. Taken from [18]

The soot deposits inside the filter and the exhaust is cleaned. Ideally, the soot should combust by itself upon landing on the filter surface, but the temperature in the filter is typically too low, and the soot particles are typically too large so a filtration cake starts to grow. This therefore leads to an increased pressure-drop, which can halt or even damage the engine if it becomes too high [18].

A Continuous Regeneration Trap (CRT) is available for large trucks

(heavy duty diesel vehicle, HDD). The trap, however, is not applicable for smaller light duty diesel (LDD) vehicles, since the temperature of exhaust gas is usually too low, and the NO_x content (the main oxidant in the CRT is NO_2, rather than O_2, as explained below) can also be too low [17].

Regeneration of the filter is therefore required and can take place through two routes:

- Non-catalyzed oxidation of soot: This is an energy consuming process, in which the filter is heated to high temperatures ($> 600°C$) [17]. An increased exhaust gas temperature is required, which can be obtained by engine controlled heating, external heating (electrical heater or fuel after burner), microwave heating, or internal heating [18]. Instead of using oxygen as the primary oxidation agent, NO_2 may be used, which can lower the temperature for the oxidation of soot to 250 - 300°C [17]. The major drawback, however, is that an oxidation catalyst is required, since most NO_x from engines are of the form of NO. As mentioned before, the amount of NO_x may be too low in some LDD vehicles for this method to work.

- Catalytic oxidation of soot: The filter may be impregnated with a catalytic compound, typically based on mixed halides and vanadates or molybdates [19]. The regeneration temperature can then be brought down to 400-450°C [20]. Another widely used method is the introduction of a fuel additive, typically cerium [16, 18]. However, ash deposits of the fuel additive oxides can build up, which may lead to plugging of the filter.

The regeneration step takes up energy from the combustion of the fuel, thereby lowering the fuel economy. A major challenge in the design of diesel particulate filters is therefore to avoid this regeneration step.

1.2.2 Particle-catalyst interaction

The problem of particle to catalyst interaction is severe in catalyzed diesel particle filters. A poor contact (often termed "loose contact") leads to almost negligible influence of the catalyst [18, 21]. Diesel particles are typically fractal-like and agglomerated of the order of 100 nm in size. Therefore, they tend to touch only a few points of the surface when deposited. Because of this, reaction is hindered by mass-transfer limitations. If a method can be developed, in which good contact between diesel soot particles and catalyst can be obtained, then this will greatly benefit the step toward a continuous catalytic regeneration trap.

Later, in section 5.4.6, the application of directly deposited films will be discussed and connections to the findings of this work will be drawn.

1.3 Deposition of nanoparticles by filtration

The principle of the formation of porous ceramic deposits by filtration is shown on figure 1.2.

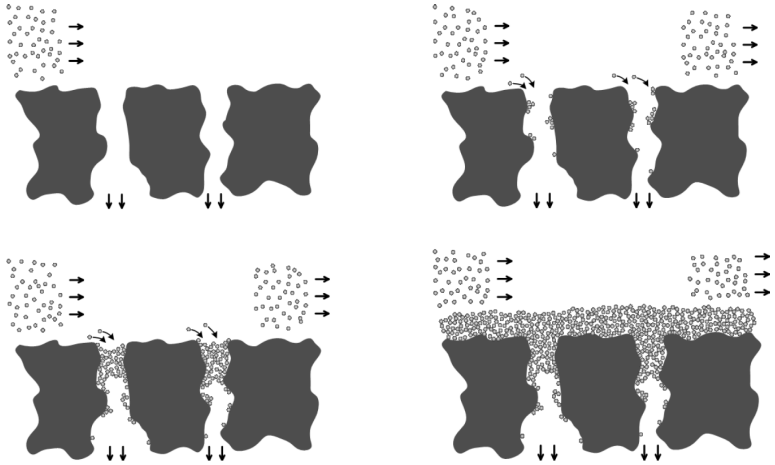


Figure 1.2 Nanoparticle deposition by filtration. A particle laden gas (upper left) is forced to pass through a substrate causing particles to deposit within and outside of the substrate. Initially, particles deposit inside the substrate capillaries (upper right). Then, a plug builds up sealing off the substrate capillaries (lower left) and finally a cake is formed on top of plug (lower right). Reprinted from [22].

A pump applies a pressure difference across a porous substrate, causing particles contained in the filtration gas to deposit. First, particles plug the porous substrate capillaries. Then, particles proceed to form a filtration cake on top of this. By controlling the flow-rate and the particle concentration, one may control the thickness of the formed deposit.

The capillary plugging process is important, as it defines the interface between the deposited material and the porous substrate, and thereby also the adhesion of the final deposit to the substrate. It has been shown experimentally [23], that after sintering such deposits tends to de-laminate from the substrate surface preserving the deposit structure intact, as shown on figure 1.3. This indicates, that the weak point is the substrate-deposit adhesion and not the intra deposit adhesion. Unfortunately, the process of capillary plugging has not been well investigated until now.

1.3.1 Current understanding of capillary plugging

The plugging process has been assumed to take place, as small particles first penetrate through the capillaries of the substrate, where they form a plug *inside* the capillary [15, 22, 24]. As plugging takes place on a limited area (substrate capillaries), the thickness of this initial deposit increases more

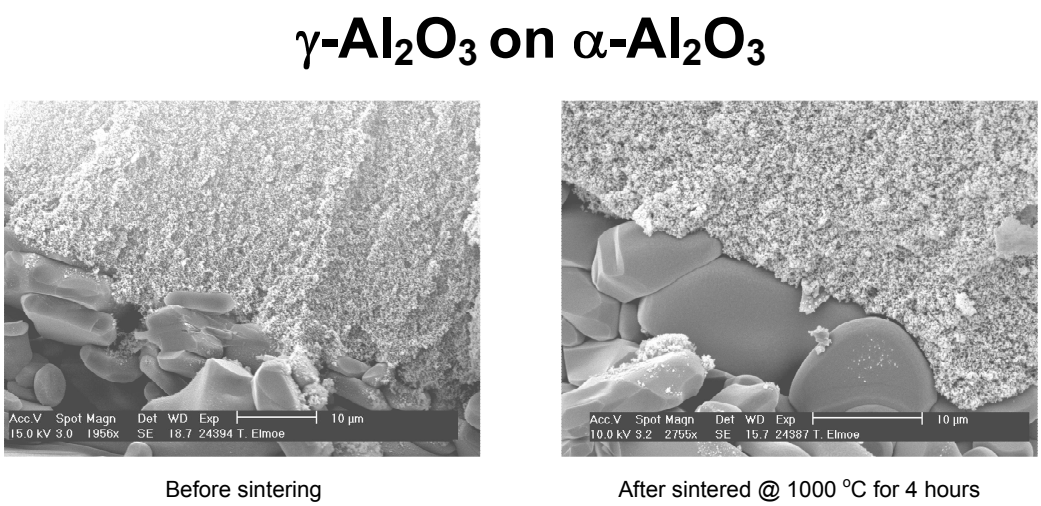


Figure 1.3 A γ-alumina cake deposited on an α-alumina substrate (left). The thermal-expansion mismatch is negligible. After sintering at 1000 °C for 4 hours, however, there is a clear gap between the cake and the substrate (right). This is most likely due to poor anchoring of the cake in the substrate capillaries.

rapidly than later on in the filtration, when a filtration cake begins to grow. Therefore, a sharp decrease of the flow-rate with time is observed. As it will be shown later in chapter 5, plugging of the capillaries does *not* take place inside the capillary itself. Although a plug is formed, it builds up on the outside of the capillary. Only a fraction of the material deposited at the time of plugging actually deposits inside the capillary.

1.4 Goals of the thesis

In order to understand the deposition of nanoparticles during filtration in detail, the project has been divided into three parts:

1. Experimental investigation of the deposition of nanoparticles by the filtration method. This is done using various characterization methods such as permeability and SEM-measurements.

2. Sintering and mechanical stability study. As the deposited agglomerates are only bound together by Van der Waals forces, the cakes are mechanically weak and sintering is applied, in order to improve the intra cake stability. The mechanical stability of the deposits is investigated quantitatively by condensing a gas in a controlled manner near its saturation pressure within the porous deposits. By doing so, the point at which they break can be measured.

3. Theoretical study of the deposition. The deposition process in a model substrate capillary is investigated using Langevin dynamics (LD). The

dependence of various parameters on the full 3D time-dependent morphology is studied. Finally, a comparison to the experimental results is given.

The applications of this deposition technique are manyfold. Due to the limited time of the project, it was not the goal to investigate exactly such a practical application. Based on the findings of this work, however, the application of the porous deposits in soot filtration from diesel cars was proposed.

Chapter 2

Particle formation in flames

2.1 Introduction

Before any particle deposition can take place, a particle source is obviously required. In this work, a premixed flame (see section 1.1) was used to produce the aerosol, since particles produced with this type of flame have been well studied [7, 23, 24]. Furthermore, in order to investigate the deposition dynamics during filtration, a low concentration of the aerosol is required, which suits well the premixed flame.

This chapter gives a general introduction into the formation of particles in flames. The structure of these particles is discussed, with emphasis on their transport properties, which deviate from that of completely spherical particles. A characterization tool, SMPS (Scanning Mobility Particle Sizer), is introduced. Since the SMPS assumes completely spherical particles, and since the particles are expected to be agglomerated, a correction formula based on the work by Lall and Friedlander [25] is shown. Then follows a section, which shows the experimental setup, as well as the flame-conditions and, finally, the results of the characterization using SMPS are presented.

The chapter does not focus so much on the particle formation itself, but rather on the characterization of the aerosol which is to be deposited, since the morphology of the deposits will undoubtedly be related to the properties the aerosol.

2.2 Theory

2.2.1 Particle formation

When a gaseous metal-oxide precursor is combusted, particles form after undergoing distinct stages [4] as illustrated in figure 2.1. First, combustion frees the metal atoms in the precursor and oxidizes them into metal oxide molecules, which are to form the particles later on. A series of stages,

which are determined mainly by the particle number concentration and flame temperature, then takes place:

- Nucleation: Stable particles clusters form as metal atoms nucleate, due to supersaturation.
- Condensation, coalescence and coagulation: Condensation can occur on the already formed nuclei and the clusters coagulate, forming particles, which upon collision with other particles can immediately coalesce to form larger particles, due to the high temperature [26].
- Coagulation only: The coalescence of particles is highly temperature and size dependent. Therefore, the effect of coalescence wears off, when the temperature decreases and the particle size increases [27, 28]. Only coagulation occurs once this is the case. The effect of coagulation, however, also decreases due to the decrease in number concentration as agglomerates grow to become bigger and bigger.

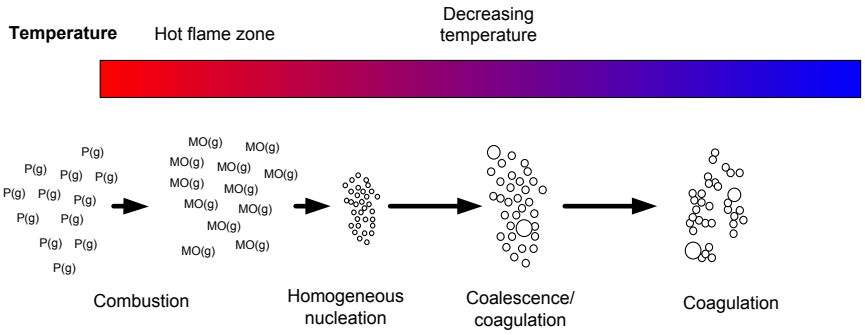


Figure 2.1 Particle formation in premixed flame with gaseous particle precursor: Combustion, nucleation, coagulation and coalescence and coagulation only. The temperature is indicated from warm (red) to cold (blue).

2.2.2 Fractal particles

If the particles experience a high temperature history, they can become coalescence limited. This means that they tend to be dense and spherical since this shape will have the lowest surface free-energy [29]. However, particles formed in flames are usually agglomerated, meaning they consists of several particles (called primary particles) forming a large coherent chain of particles [27]. The primary particles are attached to each other, either by weak van der Waals forces in chains called *soft agglomerates*, or by hard sintering necks in chains called *hard agglomerates* or *aggregates* [30]. The formation of either one is determined by the temperature profile and residence time in the flame reactor [28]. Figure 2.2 illustrates the difference between soft- and

hard agglomerates. To avoid confusion, whenever the word agglomerate is used throughout this report, it is not given whether or not this is a soft or a hard-agglomerate.

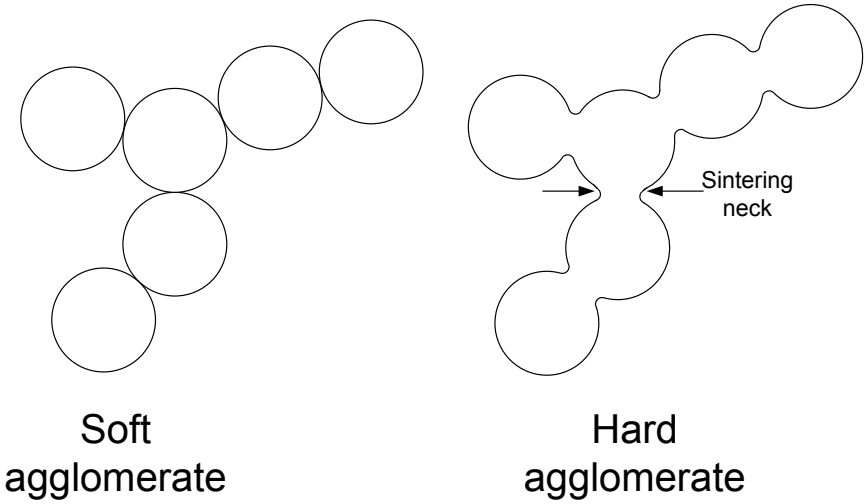


Figure 2.2 Soft- (left) and hard- (right) agglomerates. In soft agglomerates, the primary particles are held together by weak van der Waals forces. In hard agglomerates, the particles have been sintered together forming hard sintering necks.

Agglomerates are often characterized by their fractal dimension (D_f). For completely spherical and dense "agglomerates", the value of D_f is equal to the euclidean dimension (3), while for completely straight chain agglomerates, D_f becomes equal to 1 . All agglomerates that are not completely dense and spherical or straight chain-like, have a fractal dimension between 1 and 3. Equation 2.1 gives the empirical relation between the number of primary particles in an agglomerate (N_p) and the fractal dimension [29]:

$$N_p = A\left(\frac{2R_g}{d_p}\right)^{D_f} \tag{2.1}$$

where R_g is the radius of gyration, d_p is the primary particle size, A is a dimensionless prefactor typically approximated by 1 [31], however larger values have been found for highly fractal like soot- and alumina particles formed in flames ($A = 2.4$ for $D_f = 1.7$) [4].

2.2.3 Mass- and momentum transport: The Knudsen number

In particle dynamics there are three regimes which determine the proper expressions used in the calculation of mass- and momentum transport. Whether or not the particle is in a given regime is determined by the Knudsen

number (Kn). This parameter yields the ratio of the mean free length of the fluid (λ_g) to the particle size (d_p):

$$Kn = 2\frac{\lambda_g}{d_p} \tag{2.2}$$

If the Knudsen number is $\ll 1$, the particle is much greater than the mean free length and the motion of particle strongly affects the surrounding fluid. The particle acts as if it is in a continuum fluid and it is said be in the *continuum regime*. Should the Knudsen number be $\gg 1$, the particle is said to be in the *free molecular regime* as the collisions between fluid-particle does not affect the surrounding fluid and the particle effectively acts as another fluid "molecule". When the Knudsen number is ~ 1, where neither of the two different mechanisms of momentum transport are dominant, the particle is said to be in the *transition regime* [29]. The three regimes are summarized in table 2.1

Table 2.1 The three regimes for mass- and momentum transfer.

Kn	Regime
$\gg 1$	Free-molecular
~ 1	Transition
$\ll 1$	Continuum

2.2.4 Diffusion of particles

As illustrated in figure 2.3, small particles exhibit random motions due to the constant bombardment of the surrounding gas molecules on them. This effect increases with decreasing size, as the inertia transported to the particles affects more the smaller the particles. Deposition by diffusion is an important mechanism in terms of the final morphology of a deposit made by filtration of nanoparticles, as it will be shown in chapter 5. Since particles in an aerosol are not only spherical, but rather agglomerated (see section 2.2.2), calculation of the diffusion coefficient has to take this into account. However, the calculation of the agglomerate diffusion coefficient is linked to that of the diffusion coefficient for spheres, as it will be shown below in section 2.2.4.

Diffusion of spheres

The diffusion coefficient of a spherical particle of size d_p exhibiting random (brownian) motions is given by the Einstein equation:

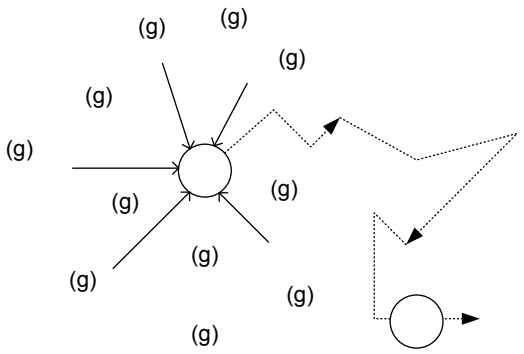


Figure 2.3 Diffusion by brownian motions. The nanoparticles exhibit a random force due to the constant bombardment from the surrounding gas-molecules, causing them to move randomly.

$$D(d_p) = \frac{k_B T}{f(d_p)} \tag{2.3}$$

where k_B is the Boltzmann constant, T is the Kelvin temperature and $f(d_p)$ is the friction factor describing the fluid drag on the particle. The friction factor factor depends on the particle size as well as on the regime at which the diffusion takes place (the Knudsen number) [29]:

$$f = \frac{3\pi\mu_g d_p}{C(d_p)} \tag{2.4}$$

where μ_g is the gas viscosity, and $C(d_p)$, is the Cunningham correction factor which corrects the friction factor for slip, and allows for the calculation of the diffusion coefficient in any of the three regimes.

In air the expression for C is [32]:

$$C(d_p) = 1 + Kn\left(1.257 + 0.4\exp\frac{-1.1}{Kn}\right) \tag{2.5}$$

Diffusion coefficient of agglomerates

The diffusion coefficient for agglomerates with low fractal dimension decreases with the size of the agglomerate, as the drag on the agglomerate increases [29]. This can cause substantial error in the calculation of the diffusion coefficient if spherical particles are assumed. For small D_f and in the free molecular regime ($Kn >> 1$), the drag force can be found by summing up the drag on each primary particle [29]. Assuming, that the agglomerate consists of N_p identical size primary particles, the diffusion coefficient of the agglomerate is then:

$$D_{\text{agg}} = \frac{k_B}{TN_p f(d_p)} \tag{2.6}$$

where $f(d_p)$ is the friction factor for one primary particle calculated from equation 2.4.

Equation 2.6 shows that the diffusion coefficient decreases with the number of primary particles in the agglomerate. For small fractal-like agglomerates this equation may hold, however for larger agglomerates, significant "shielding" of the core-particles from the outer-most placed particles may occur, and the drag becomes overestimated [33]. In this case other equations have to be used [34, 33, 35]. As it will be shown later in section 2.4, the particles produced in this work are agglomerated with a low fractal dimension. Their size allows the use of equation 2.6 in the estimation of the diffusion coefficient.

2.2.5 Aerosol characterization by SMPS

The size distribution of an aerosol can be determined through several methods. One of them is the SMPS (Scanning Mobility Particle Sizer), which counts the number of particles as a function of the mobility diameter of them. The SMPS works by first applying a known (equilibrium) charge to the aerosol using a Kr-85 source. The particles are led to a dynamic mobility analyzer (DMA) which contains a (negatively) charged center rod. The (positively) charged particles then move in the electrical field toward the rod. A fraction of them (with the right mobility) exit through a small annulus at the bottom of DMA, thereby effectively sorting them by their electrical mobility. In a condensation nucleus/particle counter (CPC) the particles are counted and a distribution based on the electrical mobility is obtained. As particles of different diameter may have the same electrical mobility, inversion techniques are required in order to determine the corresponding number distribution as a function of the mobility diameter.

The SMPS uses a sample ejector probe which dilutes and cools the aerosol to be analyzed, thereby effectively eliminating the possibility of coagulation, coalescence and sintering. Figure 2.4 shows a schematic of the ejector probe. The probe works by blowing in a large stream of (filtered) air, which causes the capillary to suck in the sample aerosol by the ejector principle. The dilution factor can be controlled by varying the capillary annulus. In this work, the capillary diameter was 1 mm and the pressure drop across the capillary was $\Delta P = 4$ bar, which yields a dilution factor of approximately 38 at room temperature. The calibration curves are shown in appendix A.1. At the sample temperature however, this factor increases due to the higher viscosity of the sample air. A method was derived to calculate the dilution rate as a function of the temperature at the probe (T_{probe}), and

it is shown in appendix A.2. The aerosol was always sampled at the top of the glass shield outlet and from the center streamline.

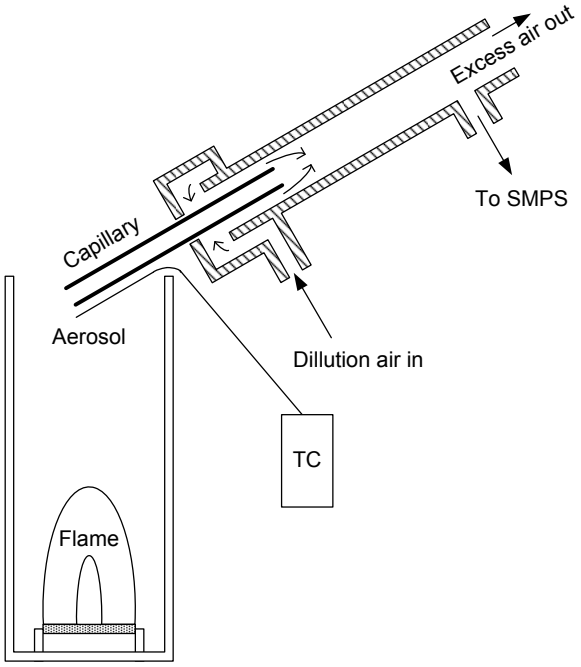


Figure 2.4 Ejector probe. Particle free air is blown in causing the probe to suck in sampled aerosol and diluted in a ratio fixed by the capillary annulus. At room temperature and at $\Delta P = 4$ bar the dilution ratio is 38. The probe samples from the upper part of the glass shield and always in the center streamline. A thermocouple (TC) measures the actual sampling temperature used to calculate the dilution ratio.

The SMPS software calculates the number size-distribution assuming the aerosol particles to be spheres. As mentioned in section 2.2.2, flame-made particles are typically fractal-like and agglomerated. Therefore, the assumption of perfect spheres over-estimates the total volume. It is possible to correct the original size-distribution obtained from SMPS for agglomerated particles using the method by Lall and Friedlander [25]. This method is valid for fractal dimensions (D_f) of the agglomerates <2, which is to be expected from past studies using the premixed flame under similar conditions [36]. A brief explanation is given below.

Agglomerate distribution from original SMPS data

If the particles were agglomerated instead, they would have the same mobility as the one calculated from drag on the spheres. The number of primary particles ($N_{p,0}$) in an model chain-like agglomerate ($D_f < 2$) [1] with the

[1] By chain-like one refers to the length to width ratio, which is >> 1 in this case.

same mobility as the corresponding sphere is [25]:

$$N_{p,0} = \frac{3\pi\lambda_g d_m}{c^* r_{p,0} C(d_m)} \tag{2.7}$$

where d_m is the mobility diameter measured by the SMPS, $r_{p,0}$ is the primary particle radius, $C(d_m)$ is the Cunningham correction factor, c^* is the dimensionless drag force equal to 9.17 [37] for agglomerates with random orientation, and λ_g is the mean free path of the gas. Equation 2.7 has been derived from a force balance for one unit charge, assuming the same mobility on the agglomerates as on the spheres. For $r_{p,0} = 5$ nm particles, roughly 25 % of the agglomerates have charge +1 , while only 5 % have charge +2. Therefore, it is fair to assume that particles only obtain one charge (+1), which simplifies the conversion between the two distributions to that of a simple charge balance:

$$n_{agg} = \frac{\eta_{sphere}}{\eta_{agg}} n_{sphere} \tag{2.8}$$

where n_{agg} is the agglomerate size-distrubution, n_{sphere} is the size-distribution assuming perfect spheres, η_{sphere} is the charge distribution for spheres, and η_{agg} is the charge distribution for agglomerates. The charge distribution for agglomerates with +1 charge follow that of a Boltzmann distribution [38, 39]:

$$\eta_{agg} = \sqrt{\frac{K_E e^2}{\pi D_{qe} k_B T}} \exp\left(-\frac{K_E}{D_{qe} k_B T}\right) \tag{2.9}$$

where K_E is a conversion factor equal to $9.0 \cdot 10^9$ N·m^2/C^2 which is used when the standard SI system is applied, and e is the electronic unit charge (1 C). The parameter D_{qe} is the charging equivalent diameter [38], and for a chain-like agglomerate it is equal to [25]:

$$D_{qe} = \frac{2 r_{p,0} N_{p,0}}{\ln(2N_{p,0})} \tag{2.10}$$

Inserting equation 2.7 into 2.10 yields an expression for D_{qe}:

$$D_{qe} = \frac{6\pi\lambda_g d_m}{C(d_m) c^* r_{p,0}} \left(\ln\left(\frac{6\pi\lambda_g d_m}{C(d_m) c^* r_{p,0}^2}\right)\right)^{-1} \tag{2.11}$$

From the mobility diameter from the original SMPS data, equation 2.11 and 2.12 can be used to find η_{agg}. Wiedensohler [40] gives the sphere equivalent charge distribution. For +1 charged particles this is:

$$\eta_{sphere}(+1) = 10^{\left[\sum_{i=0}^{5} a_i \cdot \log_{10}\left(\frac{d_m}{\mathrm{nm}}\right)^i\right]} \tag{2.12}$$

The values of the coefficients a_i are given in table 2.2.

Table 2.2 Coefficients for equation 2.12 [40]

	Charge +1					
$i =$	0	1	2	3	4	5
a_i	-2.3484	0.6044	0.4800	0.0013	-0.1544	0.0320

As η_{sphere} and η_{agg} is now known from calculations and n_{sphere} is known from the original SMPS data, n_{agg} can be calculated from equation 2.8 once the primary particle radius has been found from e.g. BET. Table 2.3 sums up the necessary equations and lists the order at which they are to be calculated.

Table 2.3 Necessary equations for the calculation of the agglomerate size distribution from original SMPS obtained sphere size distribution.

Order	Parameter	Source
1	n_{sphere}	original SMPS data
2	d_m	idem
3	η_{sphere}	eq. 2.12
4	D_{qe}	eq. 2.11
5	η_{agg}	eq. 2.9
6	n_{agg}	eq. 2.8

2.3 Experimental

In the following section, the setup of the particle producing flame is shown and the results of the characterization are presented. The main focus of this work was on the understanding and modelling of the deposition of particles. For this purpose, previously reported and optimized parameters for particle production (see section 2.3.4) were used and both the preparation and characterization serve as a basis for the next chapters. As a novelty here, the SMPS method with the agglomerate corrections by Lall and Friedlander [25] were not used in this respect before.

2.3.1 Formation of nanoparticles

Figure 2.5 shows a schematic of the experimental apparatus, which has previously been used in the production and characterization of nanoparticles [23, 24, 7]. A stream of inert carrier gas (N_2) is led through a heated saturater which contains the precursor (a metal-acetylacetonate, or short: Me-acac). The precursor sublimates and saturates the gas stream, which is led to a chamber where it is mixed with methane and air. The whole mixture is then led to a flame arrestor at which point it is ignited and burned off, with particle formation ensuing.

The Me-acac can for instance be magnesium-acac or aluminum-acac, both of which are volatile at higher temperatures and sublimate without decomposition in a non-oxygen environment. By controlling the temperature as well as the flow-rate of carrier-gas past the saturaters, one can control the amount of precursor to be fed to the burner. It is also possible to create mixed oxides by mixing together two precursor streams [7]. In this work, only alumina nanoparticles were produced to miniize any thermal expansion mismatch during subsequent stabilization of the cake (see chapter 4).

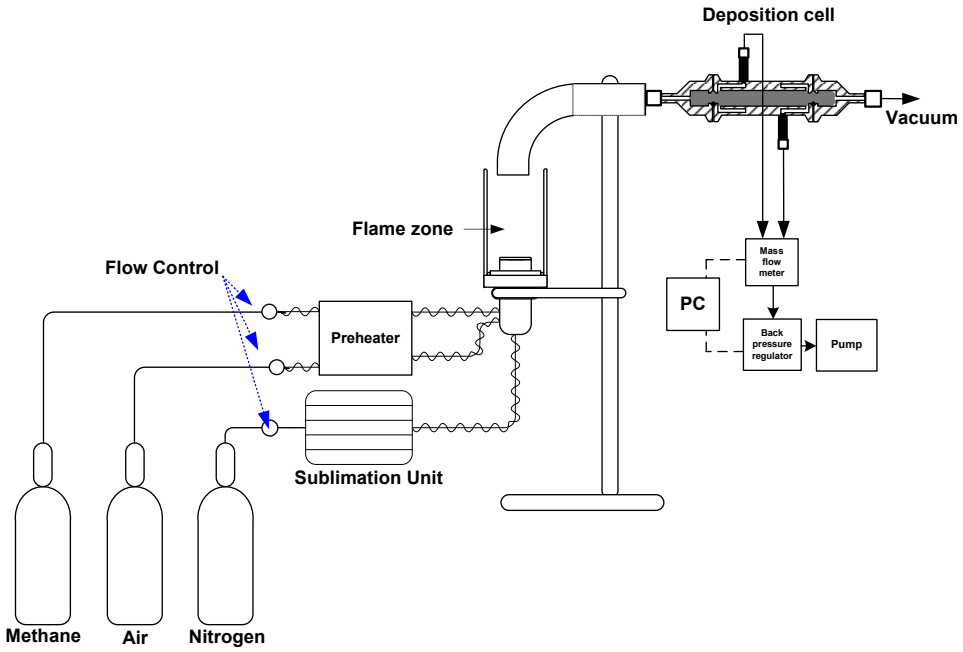


Figure 2.5 Experimental setup of the flame aerosol generator. Combustion gas (CH_4 and air) is mixed in the mixing chamber together with precursor gas (N_2 and Me-Acac), and combusted at the flame arrestor. The deposition cell is presented later in section 3.3.1.

The flame is surrounded by a quartz glass shield preventing entrainment of the ambient air, thereby keeping the mass-concentration constant.

2.3.2 Precursor properties

For the production of alumina nanoparticles, aluminum acetylacetonate ($Al(CH_3COCHCOCH_3)_3$) was used as the precursor. The physical properties, such as vapour pressure and decomposition temperatures are shown in appendix B.1.

2.3.3 Flame settings

The flame is operated stoichiometrically under the standard conditions shown in table 2.4. A quench ring is available, as it has previously been shown that by quenching the hot particle laden gas smaller primary particles can be produced [41]. Furthermore, the quench ring has also been shown to affect the crystalinity of the particles [36]. However, as the particle size-distribution of agglomerates plays an important role in the morphology of the deposits, a study on the final properties of the aerosol was done using SMPS as shown in section 2.2.5.

Table 2.4 Standard conditions for the production of alumina nanoparticles using a premixed flame.

$Q_{N_2,sat}$	Q_{Air}	Q_{CH_4}	Q_{Quench}	$h^{(1)}_{Quench}$	T_{sat}	Al_2O_3 mass conc.$^{(2)}$
(L/min)	(L/min)	(L/min)	(L/min)	cm	°C	kg/m^3
0.8	8	0.84	10	3	140	$3.27 \cdot 10^{-5}$

(1): Height above burner

(2): Calculated in appendix C.1

2.3.4 Past studies on Al_2O_3 flame-made particles

It was not the objective of this study to investigate the formation of nanoparticles, since studies on the formation of alumina particles formed in premixed flames with conditions equivalent to those used in this work already exists [7, 15, 24, 36][2]. These have shown, that the Al_2O_3 nanoparticles were either amorphous if the hot aerosol was quenched or crystalline with the γ-alumina phase crystal structure if not [36]. Furthermore, they were agglomerated with a dendritic (fractal-like) structure and had a primary particle diameter calculated from BET surface area measurements between 4.9 - 9.9 nm, depending on the precursor concentration and the flame conditions. Table 2.5 shows the mass-concentrations, as well as the primary particle diameter for several flame conditions used in the past studies.

[2]The flame-setup of Johannessen [36] and Andersen et al. [15], was modified to introduce the precursor through a free-jet.

Table 2.5 Past studies on alumina flame-made particles, under similar flame setups as in this work.

Reference	Al_2O_3 mass concentration	Total gas flow	Quenching	d_{BET}
-	kg/m^3	L/min	Yes/No	nm
[36]	$5.80 \cdot 10^{-4}$	16.43	Yes	8.1
[36]	$1.33 \cdot 10^{-4}$	11.49	No	9.9
[36]	$1.08 \cdot 10^{-4}$	14.09	Yes	9.4
[15]	$8.20 \cdot 10^{-6}$	12.53	No	4.9
[15]	$3.36 \cdot 10^{-5}$	12.53	No	6
[15]	$1.20 \cdot 10^{-4}$	12.53	No	7.1
[7]	$1.08 \cdot 10^{-5}$	11.55	No	7.1
[7]	$4.80 \cdot 10^{-5}$	11.55	No	7.9
[7]	$1.11 \cdot 10^{-5}$	11.55	No	8.8

The concentration of the aerosol in this work is in the intermediate range ($3.27 \cdot 10^{-5}$ kg/m^3) compared to that of the previous works. Therefore, the primary particle size is expected to be small. In this study, no TEM images were recorded. Attempts to determine the particle size with the BET method did not yield reasonable results, since the particles were deposited on ceramic filters, which had a surface area themselves. However, based on the previous studies performed on the flame-synthesized alumina, an estimate of the primary particle size of 8 nm was assumed for model calculations.

2.4 SMPS measurements

Figure 2.6 shows the comparison between the particle size-distribution for spheres and agglomerates, for an aerosol formed with the standard conditions shown in table 2.4. The BET primary particle radius was assumed to be 4 nm, based on past studies on similar flames (cf. 2.3.4).

The particle size-distribution estimated from the original SMPS data based on perfect spheres yields higher concentrations than for the one estimated from the agglomerates, in good agreement to what is expected for similar SMPS studies on agglomerated particles [42].

For a flame running with the settings of table 2.4 the aerosol mass-concentration is $3.27 \cdot 10^{-5}$ kg/m^3 according to table 2.4. Figure 2.7 shows the cumulated mass-distributions calculated assuming either perfect spheres or perfect agglomerates. Notice that particles below $\approx$ 13 nm can not be measured by the SMPS, which means that the calculated cumulated mass is not the complete mass of the aerosol. However, as only the smallest particles are missing from the distribution, this mass is negligible.

The assumption of perfect spheres over-estimate the total mass by 298%.

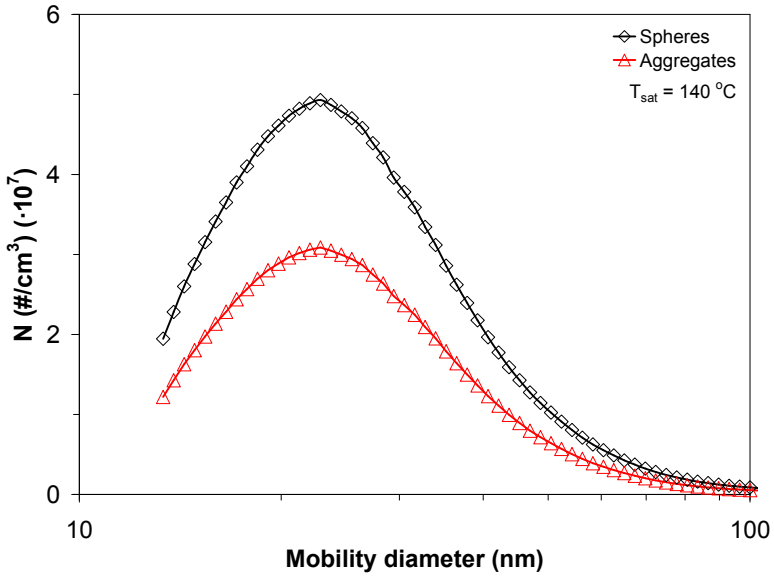


Figure 2.6 SMPS data converted to agglomerate mobility size-distribution using the method of Lall and Friedlander [25]. The flame settings can be seen in table 2.4. The data have been corrected for dilution at $T_{probe} = 400$ °C.

If perfect agglomerates (chain like) are assumed, then the estimate is smaller

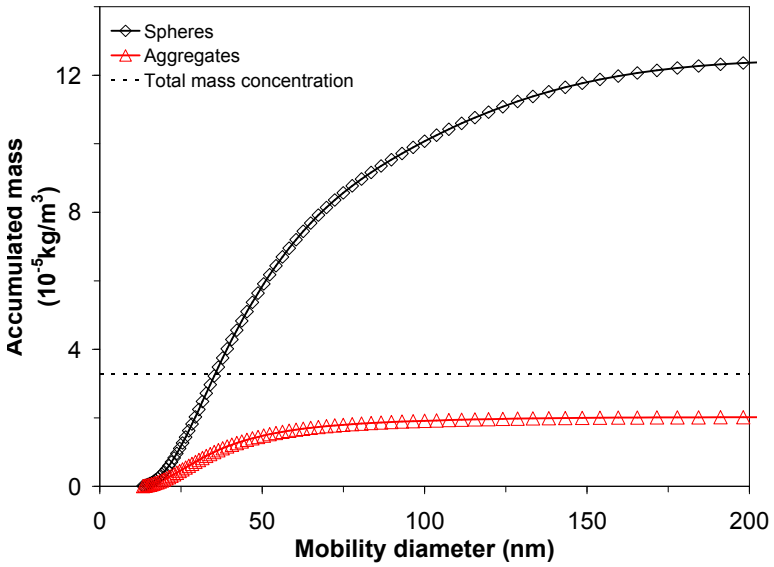


Figure 2.7 Accumulated mass-concentration vs mobility diameter. The assumption of spherical particles (black line, diamonds) clearly over-estimates (by 298%) the total mass-concentration (black broken line). For the perfect agglomerates calculated using the method of section 2.2.5, this is underestimated, however it is much closer to the total mass-contration (38% difference).

than total mass-concentration, however the difference is only 38%. Therefore, this shows qualitatively, that the particles are agglomerated and that SMPS data for such flame-made materials have to be corrected for agglomerated particles. The mass- and number-averaged agglomerate mobility diameters are found to be $\bar{d}_n = 27.9$ and $\bar{d}_m = 45.5$ nm respectively.

2.5 Conclusions

Alumina nanoparticles were produced by combustion of aluminum-acetylacetonate vapours in a premixed flame and the resulting size-distribution was characterized using SMPS. The method of Lall and Friedlander [25] was used to correct existing SMPS data for agglomerated particle. The original SMPS data overestimated the total mass-concentration by 298 %, while better comparison was obtained after correcting for agglomerates (38 % difference). This showed qualitatively, that the particles in the aerosol were agglomerated, with a number-average agglomerate mobility size of 27.9 nm.

Chapter 3

Particle deposition

3.1 Introduction

The characterization and application of flame-made nanoparticle deposits has been mostly studied on flat and non-porous substrates [10, 11, 13]. On the other hand, deposition by filtration through porous media has also been the topic of many studies [43, 44, 45, 46, 47]. Little emphasis, however, has been put on the characterization of these deposits, as the purpose of these studies was on the removal of unwanted particles.

Recently, the deposition by filtration of nanoparticles has been studied for the application as membrane filters [15, 22, 23, 24]. The deposits were characterized by SEM and by the permeability method (see section 3.2.4), but under the assumption that the cake capillaries were nanostructured (i.e. < 10 nm). Therefore, the primary mechanism for mass-transfer in the cake was assumed to be Knudsen diffusion. The resulting capillary sizes were found to be in the order 1-10 nm [23, 24]. However, the analysis has to be modified, as the high porosities, which were found in these studies (> 95 %), are detrimental to a capillary size, which is less than 10 nm (see section 3.2.5). Therefore Knudsen diffusion is not likely to be the limiting mass-transfer mechanism, and mass-transfer by viscous flow (d'Arcy flow) can no longer be ruled out.

Here, a detailed study of the morphology dependency on two process parameters, namely deposited mass and particle Pe number is performed out. For this purpose, both the present state-of-the art theory and the characterization tools are introduced. Characterization by the permeability method, is carried out without any assumption of a dominant mass-transfer mechanism, and the differences arising from the analysis in this work to that of past works shall be discussed. Finally, the possibilities, as well as the limitations of using the permeability method for characterization are discussed.

3.2 Theory

3.2.1 Dynamics of particle filtration

Particle deposition during filtration occurs through several different regimes, reflected in a change of either the pressure-drop for constant flow filtration, or the flow-rate for constant pressure-drop filtration. However, before considering the different filtration regimes for porous ceramic substrates, such as the one shown on figure 3.1a, it is necessary to introduce the "nuclepore" filter. Nuclepore filters (fig. 3.1b) consists of a thin polymer material ($\sim$ 10 μm), with a very well-defined capillary structure consisting of straight cylindrical capillaries with a narrow distribution. Due to their homogeneity they are often used in modeling studies, for the understanding of the filtration process. Spurny et al. [48, 49] studied the pressure drop evolution for

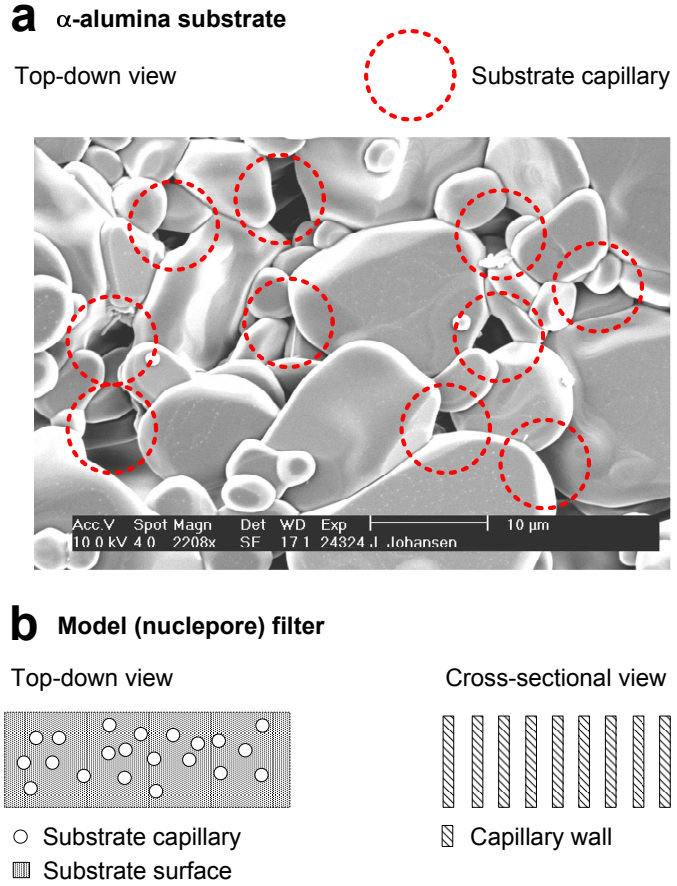


Figure 3.1 Two different filter substrates. The porous α-alumina substrate (SEM picture, a) consisting of sintered grains and the nuclepore filter (schematic, b) consisting of straight cylindrical capillaries.

constant flow-rate filtration of particles in nuclepore filters and found that

filtration takes place through several distinct regimes, as shown on figure 3.2. It is possible to extend the analysis by Spurny et al. [48, 49] to the substrates used in this work, as the principle behind filtration is the same.

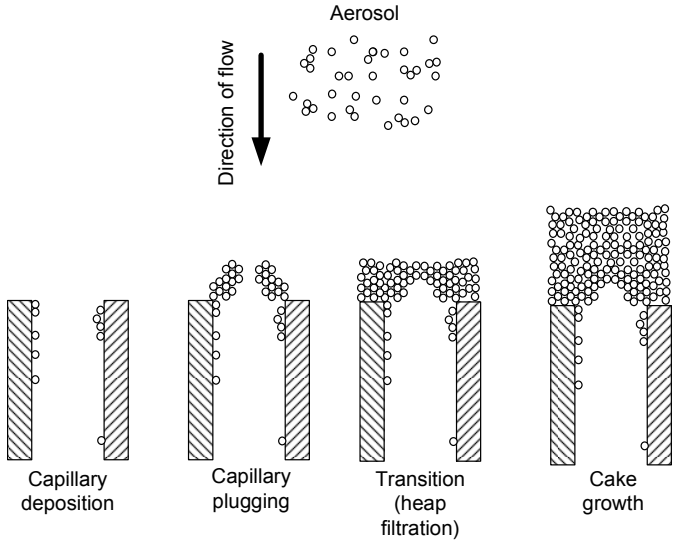


Figure 3.2 The four deposition regimes during filtration of aerosol particles.

Figure 3.3 shows a typical filtration curve (flow vs. deposited time) for constant pressure-drop filtration obtained in this work with a porous α-alumina substrate. No details about the deposition conditions are required, as the figure is only used here qualitatively.

Close observation (fig. 3.3 insert) reveals 4 regimes in the time-evolution of the flow. However, only 3 of them are clearly visible on the full time-scale. Using the nomenclature of figure 3.2, the 4 regimes are:

1. Capillary deposition.
2. Capillary plugging.
3. Transition regime (or heap filtration).
4. Cake growth.

Capillary deposition

At the start of filtration, particles deposit inside the porous substrate capillaries. This "coating" process does not significantly add to the flow resistance, since only a small amount of material deposits within the capillary. Therefore, the flow rate remains approximately constant in this regime. Due to the high concentration of particles, and the fact that capillary deposition rapidly changes into capillary plugging, this regimes is indistinguishable to

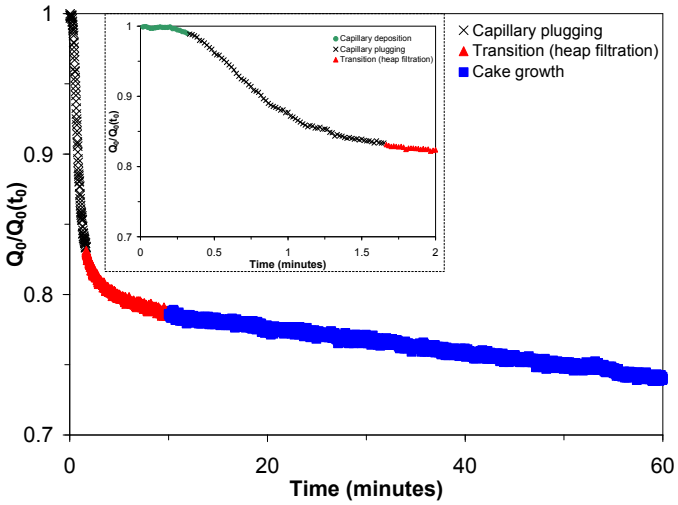


Figure 3.3 Typical filtration curve showing the different regimes of filtration, obtained during deposition by filtration of alumina nanoparticles on an α-alumina substrate. The insert shows the initial filtration curve in a higher time-resolution, revealing a flat part of the flow vs. time curve.

the time-delay from when the precursor supply is turned on to the actual point of measurement at the mass-flow meter. That this filtration regime does indeed exist will be shown in chapter 5.

Capillary plugging

After capillary deposition capillary plugging follows, as particles now deposit more and more outside of the substrate capillaries. This forms a plug of low permeability around the substrate capillaries. The flow is forced to pass through this plug which forms fast due to an increased particle flux, as the area which is deposited on here is smaller than later on in the filtration. This causes a rapid decrease of the flow-rate, which also decreases the flux of particles to the surface, and decelerates the deposition.

Transition regime (heap filtration)

Once the plug has been formed, particles continue to filter on top of the plug, forming a growing heap. Since a larger area is now available, and since the flux of particles continuously decreases due to the decreasing flow-rate, the layer forms slower than the capillary plug. The consequence is that the flow-rate decreases more slowly, and the "knee" on the filtration curve (fig. 3.2) appears.

Cake growth

After the area between the substrate capillaries has been filled up with particles, filtration now occurs on the entire surface and a filtration cake starts to grow. This occurs even slower than for the heap filtration, and is observed as a slow decrease of the flow-rate with time.

3.2.2 Deposition mechanisms during filtration

The manner in how deposition takes place depends mostly on the forces acting on the particle. Since the way a particle is deposited influences the morphology of the deposit (and by that the possible application), it is important to understand how this process can be influenced. The deposition and capture mechanisms include inertial displacement, interception and diffusional and convective deposition.

Deposition by inertial displacement

When a gas passes an obstacle, the stream-lines will naturally align themselves so that they avoid the obstacle. If particles are carried with the gas, they can be displaced from the stream-lines and collide into the obstacle due to their inertia. This is illustrated on figure 3.4.

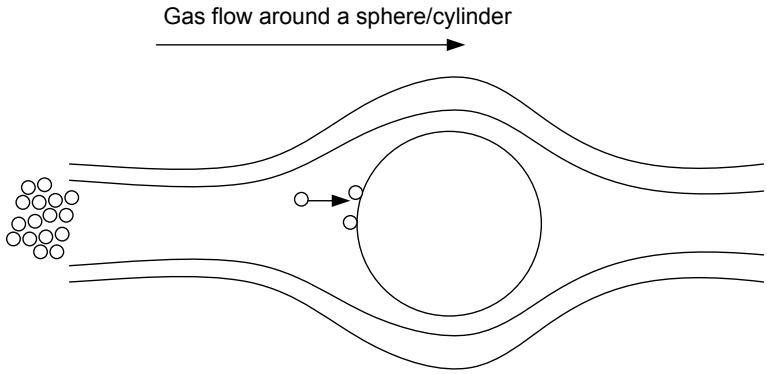


Figure 3.4 Flow around a sphere/cylinder. Due to the inertia of the particles carried by the gas, they may be displaced from the streamlines and deposited onto the object as shown.

The dimensionless Stokes number, St (here written for perfect spheres), is a quantity which defines the likelihood at which a particle is deposited by inertial impaction. It is defined as:

$$St = \frac{\rho_p d_p^2 U_0 C(d_p)}{18 \mu_g L_c} \tag{3.1}$$

where ρ_p is the particle density, U_0 is the fluid velocity, and L_c is a characteristic length of the flow geometry [50]. For small Stokes numbers ($St << 1$), impaction does not occur, while for large Stokes numbers, $St >> 1$ it does. Figure 3.5 shows the Stokes number for spherical particles with a diameter between 25 nm to 200 nm for two gas velocities (0.1 and 1 m/s, the former of which correspond to a typically obtained filter face-velocity during the deposition experiments). The characteristic length L_c has been chosen to be 1 particle radius, which corresponds to the flow around one particle already deposited.

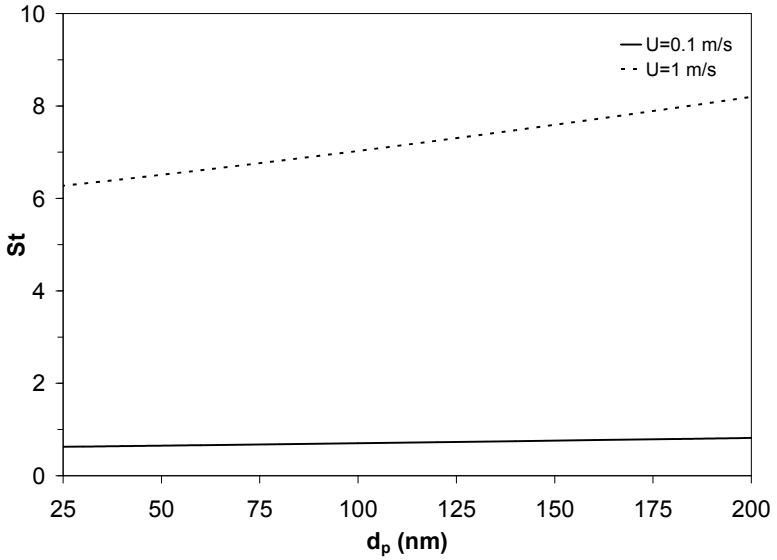


Figure 3.5 Stokes number as a function of particle diameter. Gas temperature is 200°C and gas velocity is 0.1 m/s (thick line) and 1 m/s (punctured line). The characteristic length has been taken as $L = r_p$, which corresponds to flow around a cylinder of radius r_p (one particle radius).

As it can be seen from figure 3.5, the Stokes number is not $>> 1$ or $<< 1$ for this range of particle diameters. The number-average agglomerate mobility size for the particles produced in this work was found to be 27.9 nm (see section 2.4), which yields a St number of approximately 0.6 for $U_0 = 0.1$ m/s. Therefore, deposition by impaction is expected to be low.

Capture by interception

Interception occurs when a particle passes by an obstacle within the collision range of that obstacle, hits the object, and sticks there. Interception is more of a capture mechanism rather than a deposition mechanism as it can occur regardless of the mechanism transporting the particle to the point of interception.

Convective diffusion

Pure diffusional deposition only occurs in stagnant systems and the deposition flux is rather low. The most dominant transport in filtration occurs through convective diffusion, where particles flow with the filtration gas toward the filter surface. Rather than inertial displacement, which dominates only large particles, nanoparticles are displaced from their streamlines and brought to deposit by diffusion. The Peclet number, Pe, gives the ratio of the convective transport to the diffusional transport. It is defined as:

$$Pe = \frac{U_0 d_p}{2D(d_p)} \tag{3.2}$$

The higher the Pe number, the lower the diffusional contribution to the final deposit morphology. This yields a much denser cake than what can be obtained at low Pe [51, 52, 53]. For very large Pe (Pe $\rightarrow \infty$), a *ballistic* limit of the porosity of the deposit is reached, while at Pe = 0 the fractal limit of 100% porosity is reached [54]. For deposition of monodisperse spherical particles on flat non-porous substrates, the ballistic porosity has been calculated in literature and is shown in table 3.1 for several studies. Deposition of agglomerated particles is expected to yield an even higher ballistic limit [10].

In between the limiting Pe numbers, the porosity follows an S shaped curve as shown on figure 3.6. To this author's knowledge, no such studies have been carried out so far for the deposition on porous substrates. However, a detailed study was carried out in this work and the results are shown in chapter 5.

Table 3.1 Limiting porosities for ballistic and diffusional deposition of spherical monodisperse particles

Source	ϵ (Pe $\rightarrow \infty$) (Ballistic limit)
[10]	0.84
[53]	0.85
[55]	0.85
[54]	0.84

3.2.3 Characterization of the deposited films

The deposited films are characterized by three different methods:

- Filtration curves (flow-rate vs time)
- Permeability measurements and

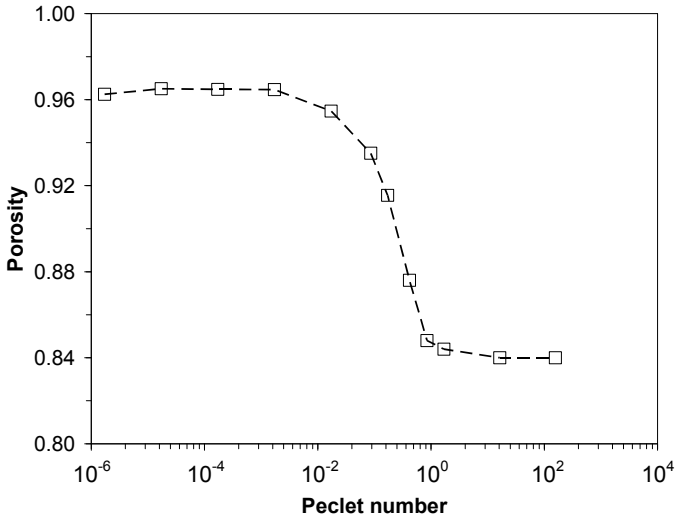


Figure 3.6 Deposit porosity as a function of the Pe number for flat substrate deposition. The porosity follows an S-shaped curve between Pe = 0 and Pe $\rightarrow \infty$. The fact that the porosity does not reach 1 at Pe = 0 is due to the finite size of the model domain [54]. Adapted from [51].

- SEM images.

The filtration curves are measured while the experiment is running. From these, information about the time evolution of the morphology can be extracted. For instance, Johansen [23] used the filtration curves to estimate the structure of the capillary plugs, assuming that a plug formed inside the substrate and that the morphology of the plug was constant (isotropic) with time. In this work, no such quantitative information was extracted, as Langevin dynamics modeling shown later in chapter 5, showed that plugging did *not* take from place inside the substrate and that the porosity of the plug was not isotropic until after plugging had occurred. The filtration curves can however be used to qualitatively identify the different deposition regimes. Furthermore, they can also be used to calculate the amount of deposited material (w_d). Typically the mass deposited before plugging is negligble in comparison to the total mass deposited, and it can be assumed that all particles are deposited on top of the substrate. The deposited mass w_d may then be estimated from a mass-balance, since the total aerosol mass-concentration (m_{tot}) is known (cf. section 2.3.3):

$$w_d(t) = m_{\text{tot}} \int_{t=t_0}^{t'=t} Q_0(t')\mathrm{d}t' \tag{3.3}$$

3.2.4 Permeability measurements

A permeability measurement is done by measuring the flow of a gas (nitrogen) across a substrate at increasing pressure difference (ΔP). By doing so before and after deposition as shown on figure 3.7a-b, the morphology of the deposits can be characterized. The substrates used in this work are porous ceramic α-alumina tubes. During the permeability measurement, nitrogen is passed through the inside of the tube and let to permeate through perpendicular to the surface of the clean substrate. After deposition, which takes place on the inner tube wall (explained later in section 3.3.1), the flow-rate decreases at the same values of ΔP due to the increased resistance of the cake.

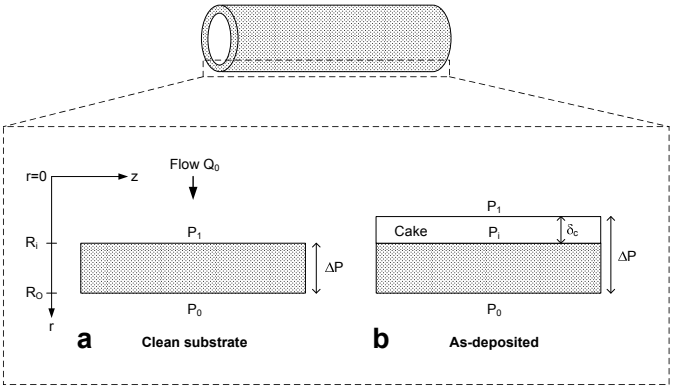


Figure 3.7 The permeability measurement and nomenclature. The flow direction is perpendicular to the surface. Before deposition (a), the flow-rate of nitrogen is measured through the clean substrate as a function of the pressure drop ($\Delta P = P_0 - P_{out}$). The deposited nanoparticles are assumed to form a cake on top of the substrate (b) and the measurement is repeated. The clean substrate measurements allows the calculation of the unknown interface pressure (P_i).

Using a mass-transfer model, it is possible to estimate values for the average capillary size of substrate and cake. Furthermore, by combining SEM and/or BET measurements with the permeability data, it is possible to estimate values for the porosity and tortuosity of the deposited layer. The chosen model is the so-called "dusty gas model" (DGM) [56], which has been successfully applied in describing the flow through multi-layered porous media [15, 57].

Dusty gas model for the clean substrate

The DGM [56] for radial flow from the inside to the outside of a clean isotropic substrate tube can be written as:

$$Q_0 P_0 = \frac{A_s}{\delta_s}\left(D_{ks} + \frac{B_{0s}}{\mu_g}\frac{P_1 + P_0}{2}\right)(P_1 - P_0) \tag{3.4}$$

where subscript s denotes the substrate, Q_0 is the flow-rate at pressure $P_0 = 1$ bar, δ_s is the thickness, D_{ks} is the Knudsen diffusion coefficient, B_{0s} is the d'Arcy permeability coefficient, $P_1 - P_0$ is the pressure-drop (ΔP) across the substrate, and A_s is the logarithmic mean area perpendicular to the flow direction:

$$A_s = \frac{2\pi L \delta_s}{\ln\left(\frac{R_O}{R_i}\right)} \tag{3.5}$$

where R_O and R_i is the inner and outer radius of the tube respectively, and L is the substrate length. Introducing the pressure-drop instead of P_1 an expression in ΔP is obtained as:

$$\begin{aligned} Q_0 P_0 =& \frac{A_s}{\delta_s}\left(D_{ks} + \frac{B_{0s}\,(2P_0 + \Delta P)}{2\mu_g}\right)\Delta P \\ =& \lambda_s \Delta P + \beta_s \Delta P^2 \end{aligned} \tag{3.6}$$

Therefore, a plot of $Q_0 P_0$ vs ΔP for the clean substrate should yield a parabolic curve with coefficients λ_s and β_s:

$$\lambda_s = \frac{A_s}{\delta_s}\left(D_{ks} + \frac{B_{0s} P_0}{\mu_g}\right) \tag{3.7}$$

$$\beta_s = \frac{A_s}{\delta_s}\frac{B_{0s}}{2\mu_g} \tag{3.8}$$

Assuming cylindrical substrate capillaries [56, 58] the Knudsen diffusion coefficient and d'Arcy permeability can be expressed as:

$$D_{ks} = \frac{d_{cs}}{3}\sqrt{\frac{8RT}{\pi M_g}}\left(\frac{\epsilon}{\tau}\right)_s \tag{3.9}$$

and

$$B_{0s} = \frac{d_{cs}^2}{32}\left(\frac{\epsilon}{\tau}\right)_s \tag{3.10}$$

where d_{cs} is the substrate average capillary size, R is the gas-constant, M_g is the molar mass of the permeating gas (nitrogen), and $\left(\frac{\epsilon}{\tau}\right)_s$ is the porosity-tortuosity factor of the substrate. Dividing 3.7 with 3.8 and inserting 3.9

and 3.10 into the resulting equation, an expression for d_{cs} can be obtained:

$$d_{cs} = \frac{64\mu_g\sqrt{\frac{8RT}{\pi M_g}}}{3\left(\frac{\lambda_s}{\beta_s} - 2P_0\right)} \tag{3.11}$$

With d_{cs} known, the porosity-tortuosity factor can also be estimated as:

$$\left(\frac{\epsilon}{\tau}\right)_s = \frac{64\beta_s\mu_g\delta_s}{A_s d_{cs}^2} \tag{3.12}$$

Dusty gas model for the substrate and cake

The DGM for a two-layered porous structure (see figure 3.7b), with isotropic structure is given below in equation 3.13 and 3.14.

For the cake:

$$Q_0P_0 = \frac{A_c}{\delta_c}\left(D_{kc} + \frac{B_{0c}}{\mu_g}\frac{P_1+P_i}{2}\right)(P_1 - P_i) \tag{3.13}$$

where subscript c denotes the cake, δ_c is the thickness, D_{kc} and B_{0c} is the Knudsen diffusion and d'Arcy permeability coefficient respectively, and $P_1 - P_i$ is the pressure-drop across the cake. For a thin cake ($\delta_c \ll R_i$), A_c can be approximated by: $A_c = 2\pi(R_i+\delta_c)L \approx 2\pi R_i L$, where R_i is the inner radius for the substrate.

$$Q_0P_0 = \frac{A_s}{\delta_s}\left(D_{ks} + \frac{B_{0s}}{\mu_g}\frac{P_i+P_2}{2}\right)(P_i - P_2) \tag{3.14}$$

By solving eq. 3.14 for the interface pressure (P_i) the following non-linear expression is obtained:

$$P_i = -P_0K_s + P_0\sqrt{(1+K_s)^2 + 2K_s\frac{Q_0\delta_s}{A_sD_{ks}}} \tag{3.15}$$

where $K_s = \frac{\mu_g D_{ks}}{B_{0s}P_0}$. A Taylor expansion to the first degree in ΔP allows the simple calculation of several parameters:

$$P_i \approxeq P_0 + \alpha\Delta P \tag{3.16}$$

The parameter α is determined by fitting P_i vs. ΔP to a straight line. The linearity of equation 3.15 holds well for low pressure drops (< 1 bar), however at higher pressure-drops equation 3.16 under-predicts the interface pressure. Keeping this in mind, the average pressure through the cake ($\frac{P_1+P_i}{2}$) can be simplified by equation 3.16:

$$\frac{P_1 + P_i}{2} = \frac{\overbrace{P_0 + \Delta P}^{P_1} + \overbrace{P_0 + \alpha\Delta P}^{P_i}}{2} = P_0 + \frac{1}{2}(1+\alpha)\Delta P \tag{3.17}$$

while the pressure drop across the cake ($\Delta P_c = P_1 - P_i$) becomes:

$$\Delta P_c = P_1 - P_i = P_0 + \Delta P - P_0 - \alpha\Delta P = (1-\alpha)\Delta P \tag{3.18}$$

Inserting eq. 3.17 and eq. 3.18 into eq. 3.13 gives:

$$Q_0 P_0 = \frac{A_c}{\delta_c}\left(D_{kc} + \frac{B_{0c}}{\mu_g}\left(P_0 + \frac{1}{2}(1+\alpha)\Delta P\right)\right)(1-\alpha)\Delta P \tag{3.19}$$

which rearranged yields:

$$\begin{aligned} Q_0 P_0 &= \frac{A_c}{\delta_c}(1-\alpha)\left(D_{kc} + \frac{P_0 B_{0c}}{\mu_g} + \frac{B_{0c}}{2\mu_g}(1+\alpha)\Delta P\right)\Delta P \quad \text{or} \Rightarrow \\ &= \lambda_c \Delta P + \beta_c \Delta P^2 \end{aligned} \tag{3.20}$$

Again, it can be seen that a plot of $Q_0 P_0$ vs ΔP yields a parabolic curve (see eq. 3.6). However, the coefficients are this time given by the structure of the cake as well as the substrate (through α):

$$\lambda_c = \frac{A_c}{\delta_c}(1-\alpha)\left(D_{kc} + \frac{P_0 B_{0c}}{\mu_g}\right) \tag{3.21}$$

and

$$\beta_c = \frac{A_c}{\delta_c}(1-\alpha)(1+\alpha)\frac{B_{0c}}{2\mu_g} \tag{3.22}$$

Following the same method as for the clean substrate, the expression for the cake capillary size d_{cc} can be found to be:

$$d_{cc} = \frac{64\mu_g\sqrt{\frac{8RT}{\pi M}}}{3(1+\alpha)\left(\frac{\lambda_c}{\beta_c} - \frac{2P_0}{1+\alpha}\right)} \tag{3.23}$$

The porosity-tortuosity factor may then be estimated as:

$$\left(\frac{\epsilon}{\tau}\right)_c = \frac{64\beta_c\mu_g\delta_c}{A_c d_{cc}^2(1-\alpha^2)} \tag{3.24}$$

The expression for the porosity-tortuosity factor is not as simple as for the clean substrate, since knowledge of the thickness of the cake is required. Typically this is only known when SEM is available. However, the deposited mass (w_d) is known from the filtration curve (see equation 3.3). For an isotropic cake, the thickness of the cake with porosity ϵ_c is then:

$$w_d = (1 - \epsilon_c)\, \delta_c A_c \rho_p \Rightarrow$$

$$\delta_c = \frac{w_d}{\rho_p A_c\, (1 - \epsilon_c)} \tag{3.25}$$

As for the thickness, the porosity ϵ_c is typically not known unless SEM images have been taken. Using a relation between the porosity and tortuosity factor, the porosity may be estimated from equation 3.24. For random porous structures one such widely used expression is [59]:

$$\tau_c = \frac{1}{\epsilon_c} \tag{3.26}$$

Inserting the expression for τ_c and δ_c into equation 3.24, the porosity of the cake may be estimated by solving:

$$\epsilon_c^2\, (1 - \epsilon_c) = \frac{64 \beta_c \mu_g w_d}{A_c^2 d_{cc}^2 \rho_p (1 - \alpha^2)} \tag{3.27}$$

Note, that the determination of ϵ_c is subject to possible errors for thin cakes, as discussed further on in section 3.2.4.

Use of the hydraulic diameter to determine the porosity

The hydraulic diameter (D_h) is an equivalent capillary diameter often used in the modeling of flow through porous media [60]. For a packed bed of spheres of diameter d_p, the hydraulic diameter is equal to [61, 62]:

$$D_h = \frac{2 d_p}{3} \frac{\epsilon_c}{1 - \epsilon_c} \tag{3.28}$$

Equation 3.28 has been derived, assuming that the spheres constituting the packed bed to form cylindrical straight and parallel capillaries with the same internal surface to free volume ratio, as the spheres. Also, the spheres are assumed to be of equal size. The tortoisity factor is not included in 3.28, however it approaches unity when the porosity is high. The assumption of cylindrical capillaries was also used indirectly in the determination of the

capillary size with the permeability method and D_h may then be replaced by the measured capillary size for the cake (d_{cc}):

$$d_{cc} = \frac{2d_p}{3}\frac{\epsilon_c}{1-\epsilon_c} \tag{3.29}$$

Andersen [15] showed that equation 3.29 predicted $d_{cc}(1-\epsilon_c)$ with less than 10 % error when compared to the permeability measurements. Solving equation 3.29 for ϵ_c yields:

$$\epsilon_{c,\mathrm{h}} = \frac{3d_{cc}}{3d_{cc}+2d_p} \tag{3.30}$$

where the subscript h denotes that the porosity has been estimated from the cylindrical capillary model. The values of ϵ_c calculated from 3.30 will be compared to the ones calculated from 3.27, as well as from SEM imaging in section 3.4.

Uncertainties in the permeability method

In the derivation of the model, it is assumed that the morphology of the substrate as well as the deposit is isotropic. Later in chapter 5, it will be shown that this assumption is only valid for the cake grown after the capillaries have been plugged. Even though all deposition experiments were carried out for times larger than the plugging time, some were carried out with times comparable to the plugging time. Since the cake porosity is in reality anisotropic until the time of plugging, its determination for thin deposits is therefore subject to possible experimental error. It may best be determined from thick layers, where the isotropic and constant porosity is reached. Reproducibility is, however, high in this method. For a series of 5 repeated measurements, the typical variation of the parameters (e.g. d_{cc}, d_{cs}, ϵ_c) is only $\approx$ 6-7 % from the average value.

3.2.5 Minimum capillary size

The capillary size determines in many cases the application of the cake. If the cake is expected to act as a gas-solid catalyst, a low mass-transfer resistance due to high porosity is beneficial [12]. However, if the application of the deposit should act as a membrane for gas separation, a very low capillary size of the cake ($<$ 10 nm) is required to obtain sufficient selectivity in the separation [63]. Figure 3.8 shows the cake capillary diameter calculated from equation 3.29, as a function of the porosity for a bed of 8 nm particles, which is the typical primary particle size of flame-made alumina nanoparticles (see section 2.3.4). The minimum porosity reachable with the direct

deposition method (see table 3.1) is marked in the gray zone of figure 3.8. The broken line shows the limit of closest spherical packing ($\epsilon = 26\%$) for comparison.

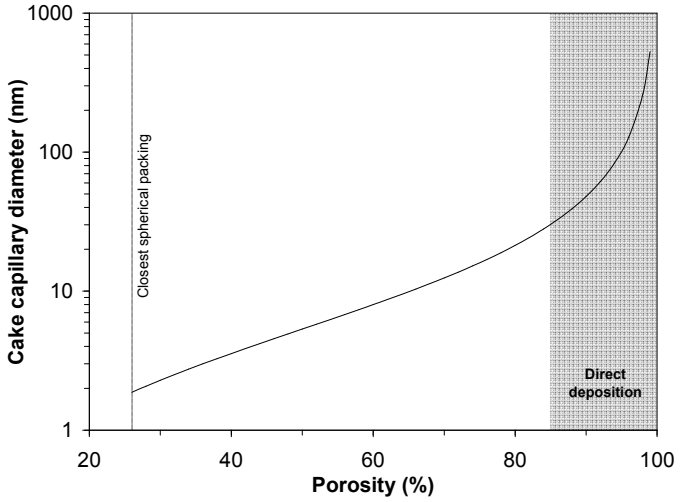


Figure 3.8 The reachable capillary size for cake deposits calculated from equation 3.29 for 8 nm particles. The range reachable with the direct deposition method is marked with as gray. The closest spherical packing is marked as the broken line at $\epsilon = 26\%$.

If ballistic deposition is possible ($\epsilon_c \approx 85\%$), then the capillary-size may be as low as 30 nm. However, it is not possible to reach $O(1)$ nm capillary diameter with the direct deposition method. In fact, the porosity should be decreased to 65% if a capillary diameter < 10 nm is required. To do so, restructuring is required. One way of restructuring the cake, is by simultaneous sintering and deposition, when the characteristic time for sintering is much smaller than the time between particle arrival to the substrate [50, 64]. This methods requires very high temperatures ($> 1000°$C) as well as a low aerosol concentration. Due to the high temperature, any thermal expansion mismatch can lead to peeling, cracking and lift-off of the deposited cake [23] (discussed later in section 4.2.6).

3.3 Experimental

3.3.1 Particle deposition

Alumina nanoparticles, produced in a premixed flame using the settings of table 2.4 and characterized in section 2.4, were deposited on a porous ceramic tube (α-alumina). Figure 3.9 shows a schematic of the deposition cell. The ceramic substrate was sealed using o-rings. By applying a low pressure on the outside of the tube, the aerosol was forced through the substrate. This caused particles to be filtered off on the inner substrate surface, and a porous

ceramic film was formed [15, 24, 22, 5, 23]. A back-pressure regulator (BPR) adjusted the pressure for constant pressure-filtration and a mass-flow meter (MFM) measured the flow rate through the substrate. A PC equipped with Labview was used for the control of the BPR and the collection of data from the MFM.

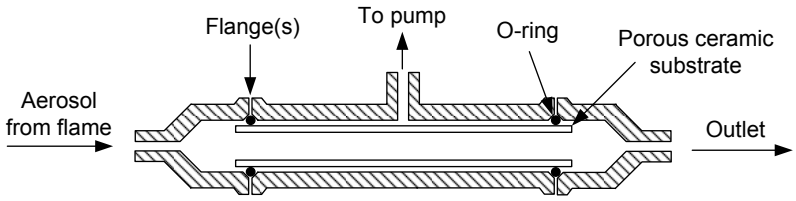


Figure 3.9 Schematic showing the deposition cell. Aerosol laden gas from the flame enters at the inlet. A pump applies a low pressure on the outside and a back-pressure regulator controls the pressure-drop across the substrate. The flow is measured by a mass-flow meter and the data is collected via Labview on a PC.

3.3.2 Substrate properties

The substrates were porous α-alumina ceramic tubes with the dimensions given in table 3.2.

Table 3.2 α-alumina substrate dimensions

Inner radius (R_i)	Outer radius (R_O)	Length (L)
mm	mm	mm
3.25	4.75	23.5 / 58

The porosity of the substrates was measured by first weighing the dry mass of the tubes, dipping them into water then weighing the wet mass, then weighing them again. Once the mass of water contained in the substrate capillaries was known, the capillary volume could be calculated. From the total volume of the substrate, the porosity could be determined from:

$$\epsilon_{s,1} = \frac{\frac{m_{\text{water}}}{\rho_{\text{water}}}}{\pi \left(R_O^2 - R_i^2\right) L} \tag{3.31}$$

where m_{water} and ρ_{water} is the mass and density of water respectively. Another method is to weigh the dry support, then calculate the expected mass from the solid density ($\rho_{\text{alumina}} = 3900$ kg/m^3) and the volume of the support:

$$\epsilon_{s,2} = 1 - \frac{\frac{m_{\text{substrate}}}{\rho_{\text{substrate}}}}{\pi \left(R_O^2 - R_i^2\right) L} \tag{3.32}$$

By the first method (equation 3.31), the entire volume accessible is determined, which includes dead-end capillaries. In the latter method (equation 3.32), the entire capillary volume is considered, which includes "closed-in" capillaries. Typical results of the permeability and porosity measurements are shown in table 3.3.

Table 3.3 α-alumina substrate properties. The average capillary radius was measured using the permeability of section 3.2.4. The porosity was calculated using equation 3.31.

Average capillary diameter* (d_{cs}) μm	$\epsilon_{s,1}^{**}$ %	$\epsilon_{s,2}^{**}$ %
4.0 ± 1.8	31.3 ± 1.12	30.6 ± 1.31

*: Average of 15 samples

**: Average of 12 samples

The maximum porosity ($\epsilon_{s,2}$) is somewhat smaller than the accessible porosity ($\epsilon_{s,1}$), however the difference is well within the experimental error as indicated in table 3.3. In fact, it can be concluded that the substrates does not contain any closed-in capillary volume.

3.3.3 Deposition conditions

Deposition conditions are shown in table 3.4. The deposition cell temperature, T_{cell}, was kept constant at approximately 200°C throughout deposition. To investigate the influence of fluid velocity on the morphology, two different pressure-drops was used. The maximum pressure-drop, which gives the maximum flow-rate through the substrate, is 0.5 bar. As it is the Pe number which determines the morphology (cf. section 3.2.2), the average Pe number experienced by most particles for flow in the substrate capillaries is also shown. This is calculated from the measured average mobility diameter of the agglomerates (27.9 nm), shown in section 2.4. The diffusion coefficient of the agglomerate is estimated using the equations of section 2.2.4. By assuming a low (< 2) fractal dimension, the number of primary particles in an agglomerate of 27.9 nm was estimated from eq. 2.6 to be 19, when the primary particle diameter is assumed to be 8 nm. This gives a diffusion coefficient of the agglomerate of approximately 61 % of that of a dense sphere with the same mobility diameter. With this diffusion coefficient and the given pressure drop (ΔP), two sets of Pe numbers can be calculated: 0.3 and 0.5. The Pe numbers are given for the initial deposition which occurs at the substrate capillaries. This shows, that deposition takes place far from the ballistic limit that is at Pe > 10 (cf. section 3.2.2), and that the morphology is expected to be dominated by the diffusional deposition.

Once plugging has finished and cake growth has begun, the fluid velocity will decrease as filtration takes place on the entire filter surface. Therefore,

the Pe number decreases during filtration and the porosity of the cake should be higher than the initial plug. However, as the Pe number is already low this effect is expected to be small.

With the substrates and the pump used in this work it was not possible to obtain a gas velocity high enough to reach Pe > 0.5. Possible means of achieving higher Pe numbers are discussed in the outlook of this work (section 6.1). In chapter 5 a detailed modelling study of the deposition during filtration is presented. There, the studied range of Pe numbers is not limited by the substrate

Table 3.4 Experimental conditions for deposition of alumina nanoparticles produced with the flame settings of table 2.4.

Experiment label	Substrate (d_{cs}) capillary size	T_{cell}	ΔP	Pe*	Deposition time	Deposited mass
-	μm	°C	bar	-	minutes	mg
tde-07	3.47	205	0.3	0.3	60	5.19
tde-08	2.95	215	0.3	0.3	30	2.01
tde-09	4.03	215	0.3	0.3	10	0.70
tde-12	3.88	210	0.3	0.3	10	0.74
tde-13	4.55	216	0.3	0.3	10	0.77
tde-16	3.36	210	0.3	0.3	10	0.76
tde-17	3.11	210	0.3	0.3	10	0.80
tde-29	2.24	210	0.5	0.5	60	8.19
tde-30	8.34	210	0.5	0.5	60	5.76
tde-31	7.82	180	0.5	0.5	60	8.66
jta-perm-18	3.72	300	0.35	0.3	480	36.7

*: The capillary Pe number was calculated from equation 3.2 assuming agglomerated particles with mobility diameter 27.9 nm, and $T = T_{\text{cell}}$.

3.4 Results and discussion

3.4.1 Visual inspection of the cake morphology

Figure 3.10 shows an SEM image for deposition experiment jta-perm-18. The cake appears to be highly porous and formed by particles clustered together into more dense regions of agglomerates, which are separated by chains of particles. The distance between the agglomerated structures gives the sizes of the cake capillaries. These seem to be in the order of 100 - 1000 nm, however only the top part of the cake was examined (fig. 3.10a). The cake capillary size may decrease down through the cake, as the porosity at the top part is expected to be the highest since deposition was halted there. A higher porosity yields a higher capillary size, as predicted by figure 3.8.

Certainly, when comparing the results of the permeability measurements (shown later in section 3.4.4), the cake capillary size which can be estimated from SEM seems too large.

The size of the dense regions are much larger ($\sim$ 1 μm), than the average agglomerate mobility size (27.9 nm) found in section 2.4. Therefore, they must have formed during deposition. This is possible, in particular for deposits done at these low Pe numbers (0.3). Such deposits are fractal-like and grow in a "tree"-like manner, as it will be shown later in chapter 5. As particles are intercepted, they shade off the area below, increasing the chance of intercepting another particle at that position. As a consequence, some regions will be more dense in particles than others.

The cake appears uniform in thickness across the substrate length, and there is a fine interface between cake and substrate (fig. 3.10b). In fact, it is difficult to see any significant particle penetration into the substrate.

The thickness of the cake is estimated to be between $\approx$ 135 μm. From the deposited mass ($w_d = 36.7$ mg) and the total filtration area, the porosity is estimated from equation 3.25, here solved for ϵ_c:

$$\begin{aligned}\epsilon_c &= 1 - \frac{w_d}{\rho_p A_c \delta_c} \\ &= 1 - \frac{36.7 \cdot 10^{-6}}{3900 \cdot 2\pi(0.00325 \cdot 0.058) \cdot 135 \cdot 10^{-6}} = 0.94\end{aligned}$$

The deposition temperature of jta-perm-18 is approximately 90°C higher than for the "tde"-denoted experiments (see table 3.4), however the temperature is still well below that at which restructuring is expected. Therefore the porosity obtained here is representative for the ones obtained for lower temperature depositions.

Figure 3.11 shows an SEM image for experiment tde-16. The cake is much thinner than the one shown on figure 3.10, and the calculated porosity (0.97) is abit larger than for jta-perm-18. The difference is however most likely within experimental uncertainties and the results agree well with previous studies [15, 24, 23].

In section 3.4.5, the SEM calculated porosity is compared to the one calculated from permeability measurements.

3.4.2 Low deposition mass: tde-08

Figure 3.12 shows the filtration curve (flow vs. deposited mass) for experiment tde-08. The deposition conditions was shown in table 3.4. Filtration was carried out for 10 minutes.

A full qualitative description of the filtration curve was given in section 3.2.1, and at least three of the four regimes shown in figure 3.3, are also seen on figure 3.12. Capillary plugging is observed as the period at which the

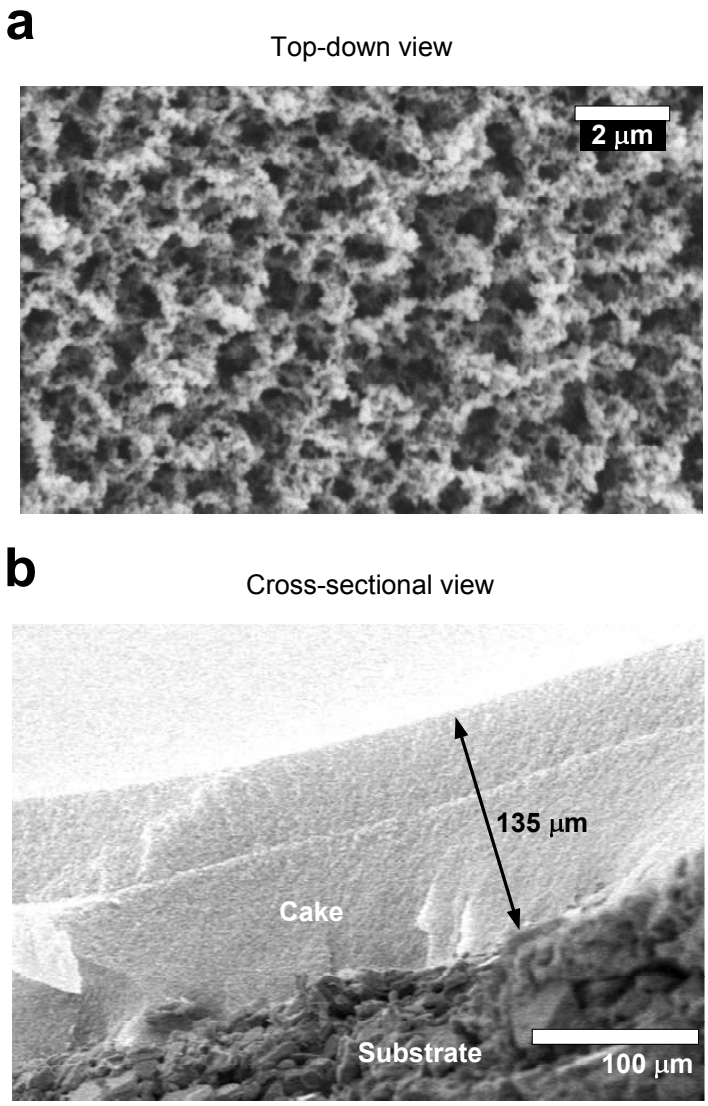


Figure 3.10 Top-down (a) and cross-sectional (b) SEM of a very thick cake deposited by the filtration method. Experiment label: jta-perm-18 (cf. table 3.4). The cake consists of dense regions of particles (agglomerates), separated by chains of particles. The distance between the dense regions as seen from the top-down view is between 100 - 1000 nm. The thickness (b) is estimated to be approximately 135 μm, and the porosity is calculated to be ≈ 0.94 (see text).

flow-rate decreases most rapidly, and is seen to occur after ≈ 0.1 mg of the aerosol has been deposited, which is within a few minutes from the start of

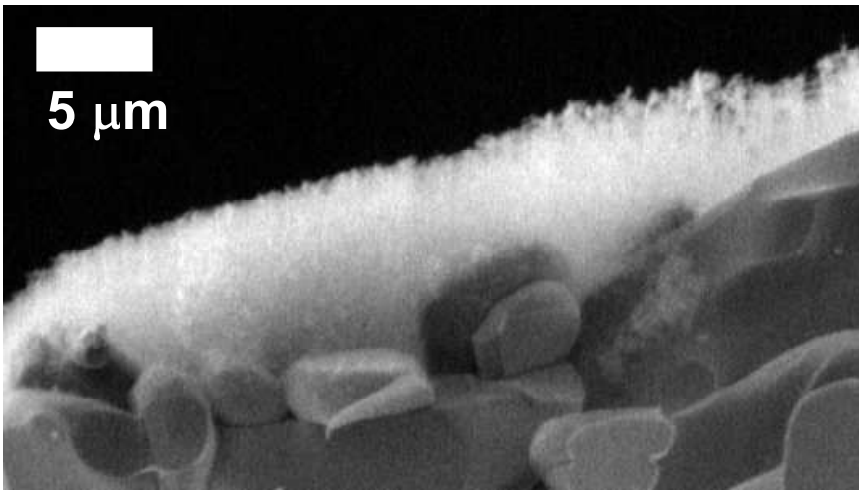


Figure 3.11 SEM of a thin cake deposited by the filtration method. Experiment label: tde-16 (cf. table 3.4). The thickness is estimated to be approximately 5 μm, yielding a porosity of 0.97.

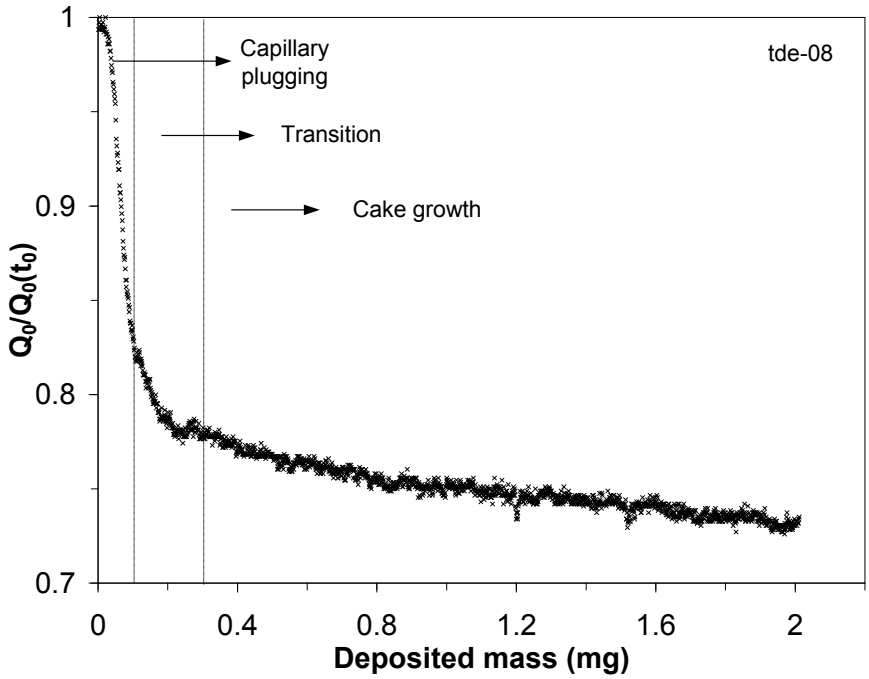


Figure 3.12 Relative change in the flow-rate ($Q_0/Q_0(t_0)$ vs deposited mass for constant pressure deposition of Al_2O_3 aerosol (experiment: tde-08). Flame conditions are listed in table 2.4. Deposition conditions in table 3.4.

filtration.

Figure 3.13 shows the permeability (flow-rate vs. pressure-drop) curve for the experiment before and after deposition. As expected from the filtration curve (figure 3.12), the permeability has decreased after deposition.

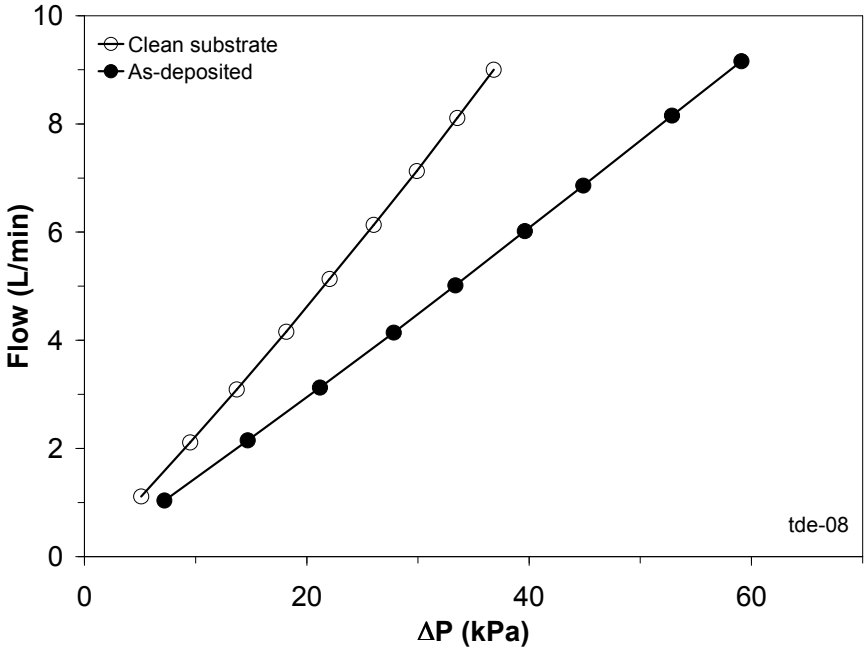


Figure 3.13 Permeability (flow-rate vs ΔP) for experiment tde-08 before and after deposition.

Applying the permeability method described in section 3.2.4 the morphology is characterized and a summary of the results is shown in table 3.5. The porosity calculated from the permeability data is very close to 1. The porosity estimated from the cylindrical capillary model is also shown for comparison. Because of the high porosity, the difference ($\frac{|\epsilon_c - \epsilon_{c,\mathrm{h}}|}{\epsilon_c}$) between the two methods becomes small, and it is more appropriate to use the solid volume fraction $\phi_{s,c} = 1 - \epsilon_c$ to compare. Using this value, the difference between the two methods then becomes 333 %. The very high porosity calculated from the permeability method and the large difference that calculated with the cylindrical capillary model, shall be discussed in section 3.4.5.

3.4.3 High deposition mass: tde-31

Figure 3.14 shows the filtration curve for the deposition experiment tde-31. The three flow regimes due to capillary plugging, transition regime, and cake growth are clearly visible. Capillary plugging occurs after approximately 0.3 mg of the aerosol has been deposited. Compared to the low mass deposition

Table 3.5 Characterization of the cake morphology for experiment tde-08.

$w_d^{(1)}$	$d_{cc}^{(2)}$	$\epsilon_c^{(3)}$	$\epsilon_{c,\mathrm{h}}^{(4)}$
mg	nm	-	-
2.01	170.7	0.992	0.970

(1): Calculated from 3.3
(2): Calculated from 3.23
(3): Calculated from 3.27
(4): Calculated from 3.30 assuming a primary particle size of 8 nm

experiment (tde-08), this is approximately three times the amount deposited before capillary plugging. However, the substrate average capillary size for tde-31 is approximately 2.7 times larger, than for tde-08, which explains why more mass should be deposited in the substrate capillaries for tde-31 before plugging.

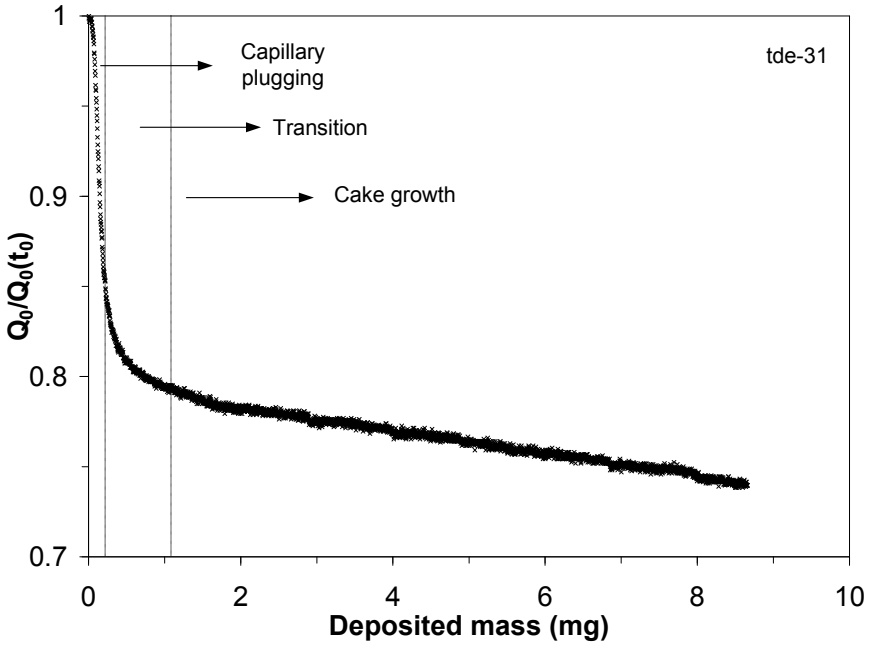


Figure 3.14 Relative change in the flow-rate ($Q_0/Q_0(t_0)$ vs deposited mass for constant pressure deposition of Al_2O_3 aerosol (experiment: tde-31). Flame conditions are listed in table 2.4. Deposition conditions in table 3.4. Deposition was carried out for 60 minutes, and the deposited mass was approximately 5 times that of tde-08.

The permeability curves before and after deposition of tde-31 are shown on figure 3.15. The results of the permeability characterization is shown in table 3.6. The porosity calculated from permeability agrees within $< 7\%$

difference to the porosity calculated from the cylindrical capillary model. The reason for this will be discussed in section 3.4.5.

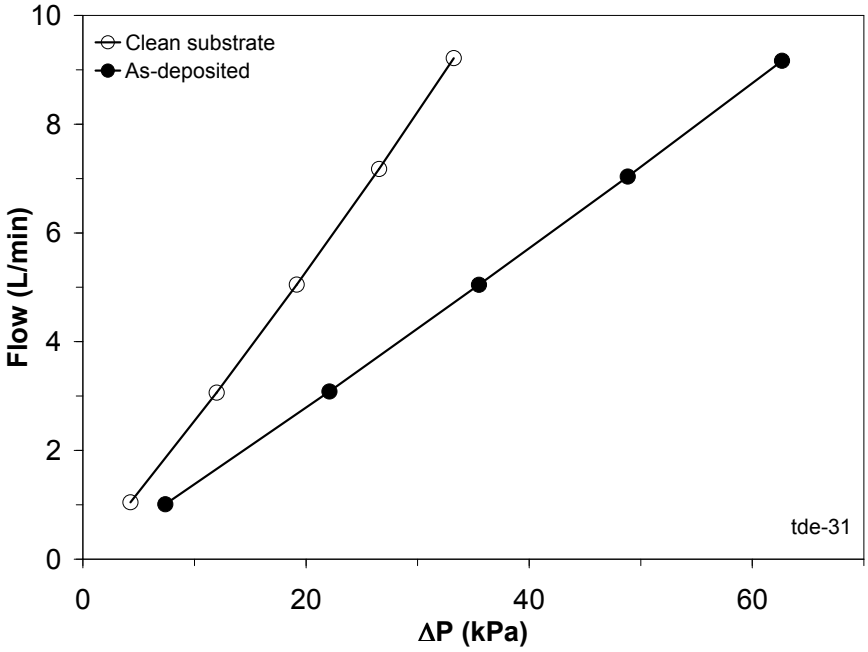


Figure 3.15 Permeability (flow-rate vs ΔP) for experiment tde-31 before and after deposition.

Table 3.6 Characterization of the cake morphology for experiment tde-31.

$w_d^{(1)}$	$d_{cc}^{(2)}$	$\epsilon_c^{(3)}$	$\epsilon_{c,\mathrm{h}}^{(4)}$
mg	nm	-	-
9.01	160.8	0.970	0.968

(1): Calculated from 3.3
(2): Calculated from 3.23
(3): Calculated from 3.27
(4): Calculated from 3.30 assuming a primary particle size of 8 nm

3.4.4 Investigation of cake capillary size

Figure 3.16 shows the variation of the cake capillary size with changing deposited mass for two sets of Pe numbers. Increasing the Pe number from 0.3 (fig. 3.16 triangles) to 0.5 (fig. 3.16 squares) does not markedly seem to decrease the cake capillary size. This is expected from modelling studies of deposition during filtration in porous substrates, presented later in chapter

5, since the Pe range investigated is too narrow for any change to be detected outside of the experimental uncertainties. Also, no clear trend is seen by increasing the deposited mass.

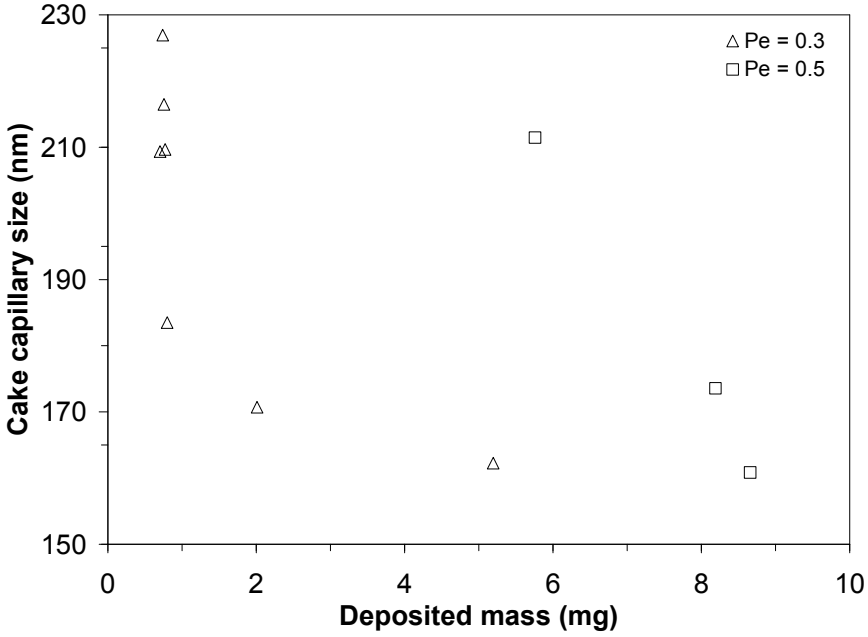
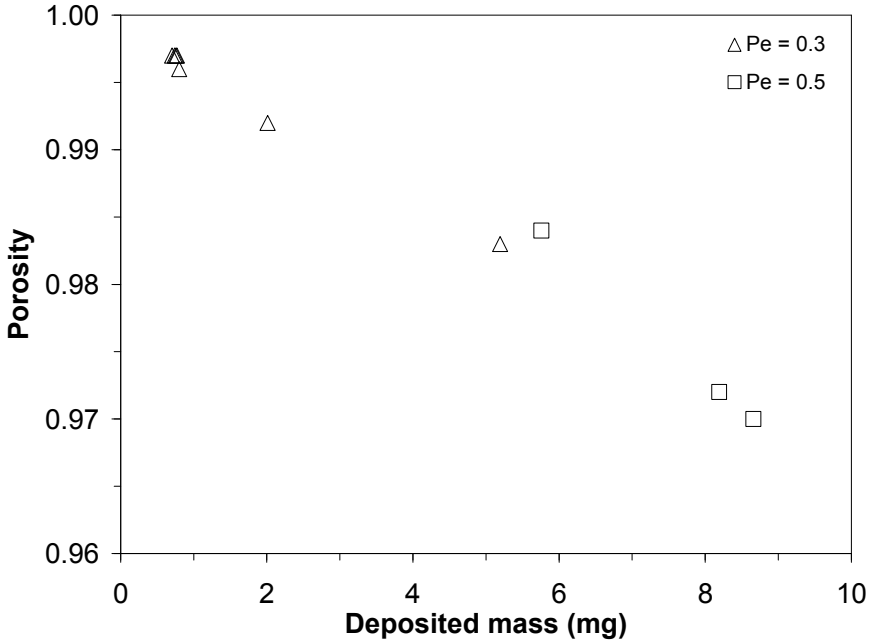


Figure 3.16 Cake capillary size variation with deposited mass for two sets of Pe numbers.

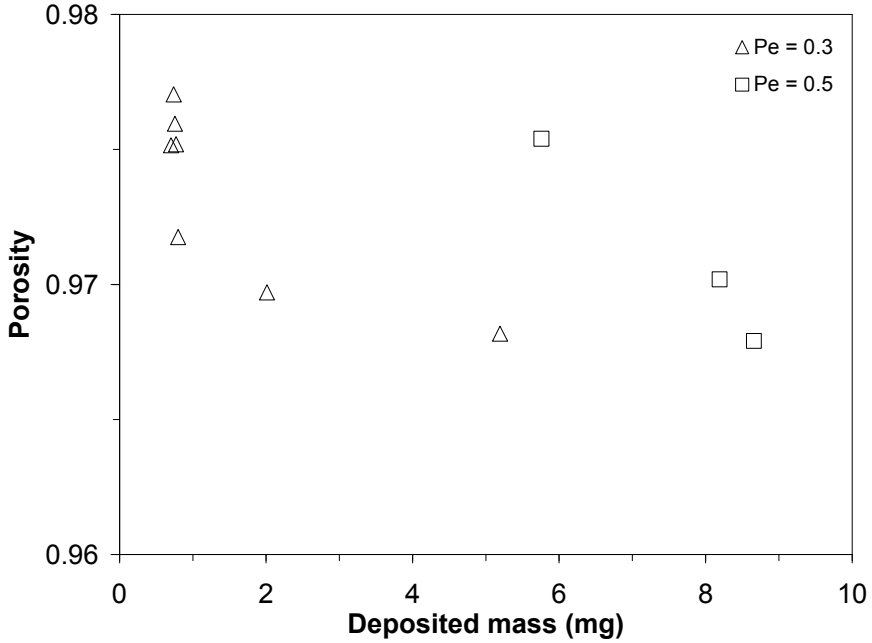
The cake capillary size (> 100 nm) is much larger, than what was found previously (O(1-10) nm) for deposition of particles during filtration with similar flame conditions [22, 24, 23]. The reason for this is most likely, that no assumptions of a dominant mass-transfer mechanism has been made. The cake capillary size calculated in this work also agrees much better with SEM images (see section 3.4.1). It is certainly hard to justify how cake capillary sizes of O(1-10) nm should limit the mass-transfer for so highly porous (> 94 % porosity) cakes.

3.4.5 Investigation of the cake porosity

As it was shown in the previous section 3.4.3, the permeability method and the cylindrical capillary model gave both comparable values for the porosity when the deposited mass was high (> 8 mg). Figure 3.17a-b shows the full evolution of the porosity with deposited mass calculated with the permeability method (a) - and cylindrical capillary model (b) respectively. Both sets of Pe numbers was used, as it was shown on figure 3.16 that increasing the Pe number from 0.3 to 0.5 did not change the cake capillary size. Therefore, it is also not expected that the porosity will change markedly in the investigated Pe range.



(a) Cake porosity determined by the permeability method as a function of the deposited mass. For low masses, the porosity is near 1, which is unrealistically high. The reasons for this is discussed in the text.



(b) Cake porosity determined by the cylindrical capillary model as a function of the deposited mass. The primary particle size was assumed to be $d_p = 8$ nm.

Figure 3.17

The porosity calculated from the permeability method (fig. 3.17a) follows a trend: at low deposited mass, the porosity is higher than later in the deposition process. However, the porosity is extremely high at low mass, which compares poorly to what is expected from SEM (0.94 - 0.97). The reason for this is most likely a wrong assumption in the permeability model: The path of the gas is always through that part which gives the least pressure-drop. For thin cakes (low deposited mass), the gas is likely to pass through above the substrate capillaries as shown on figure 3.18a. Therefore the flow cannot be assumed to be perpendicular to the total area throughout the entire cake thickness. Since the actual flow-profile may be quite complex at this scale, it is unlikely that any analytical expression can be found. An assumption of the flow-profile then has to be made. Figure 3.18b shows the equivalent flow-profile assuming, that the stream-lines have aligned themselves to only pass preferentially through the part of the cake which is above the substrate capillary. The flow area therefore changes from A_c to $\epsilon_s A_c$.

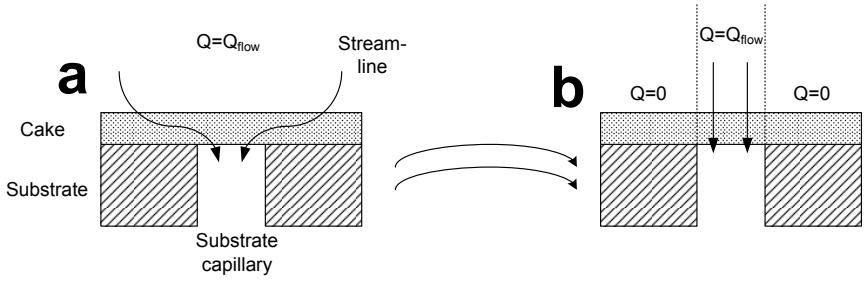


Figure 3.18 Flow-profile (a) through a thin cake. The stream-lines align themselves to the path of least pressure-drop. Significant deviation from the assumption of a perpendicular flow is expected. (b): The assumed flow profile. The gas is assumed to only pass preferentially through the part of the cake which is above the substrate capillary.

Whereas, the equation for calculating the cake capillary size remains unchanged, the equation for calculating the porosity (eq. 3.27) has to be modified. The cake thickness is determined from the entire area, since during deposition the gas will flow as shown on figure 3.18a. The modified equation 3.27 then becomes:

$$\epsilon_c^2 (1 - \epsilon_c) = \frac{64 \beta_c \mu_g w_d}{A_c^2 \epsilon_s d_{cc}^2 \rho_p (1 - \alpha^2)} \tag{3.33}$$

The assumption of a preferential flow-profile is only made in order to apply the permeability method, however the thinner the cake, the more accurate the assumption of this flow-profile becomes.

Figure 3.19 shows the corrected porosities (using eq. 3.33) determined by permeability data. Only the data for the low mass ($\approx$ 0.7 mg) deposi-

tion experiments have been corrected, since these experiments were stopped shortly after capillary plugging (10 minute deposition). The figure shows, that correcting the low deposition mass experiments for the flow-profile, brings down the cake porosity. The values are, however, still far from the SEM value, showing that the porosity is best determined for thick cakes, where a uniform flow-profile has been achieved. It should also be mentioned, that surface roughness may also play an important role, especially for thin cakes, as the asperities on the surface can be up to 10 μm (size of one substrate grain). Therefore, the assumption of a cake of uniform thickness may not be entirely correct when the thickness is comparable to the surface asperities.

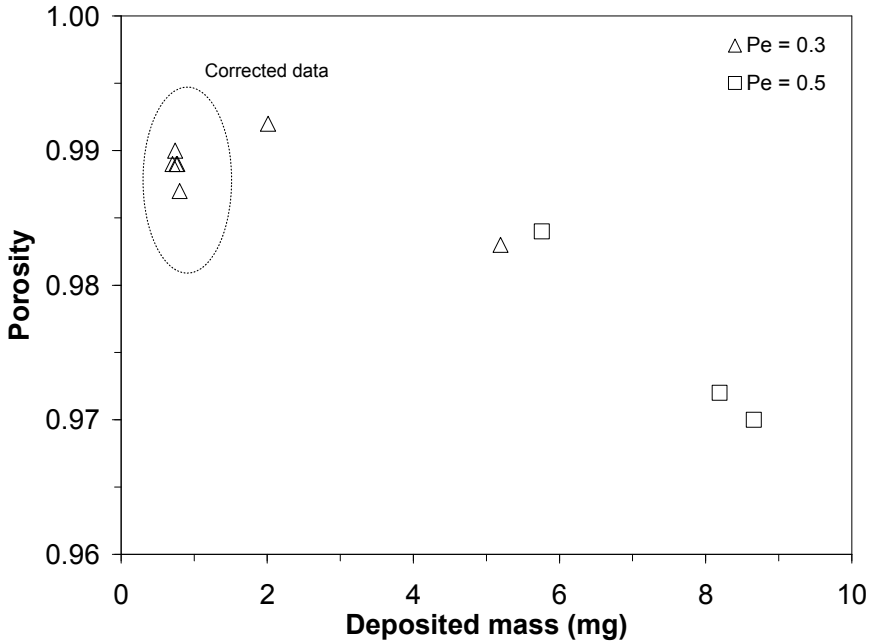


Figure 3.19 Porosity calculated using the permeability method as a function of the deposited mass. The low mass deposition experiments have been corrected using eq. 3.33.

Opposite to the permeability method, the porosity calculated from the cylindrical capillary model (fig. 3.17(b)) does not follow any trend with the deposited mass. This is because the cake capillary size, which is independent on the deposited mass (see fig. 3.16), is used in the cylindrical capillary model for the calculation of the porosity (see eq. 3.30).

3.5 Conclusions

Highly porous ceramic cakes consisting of alumina nanoparticles were formed on porous α-alumina substrates by filtration. At least three regimes were

identified from the shape of the filtration curve: *capillary plugging*, *transition regime*, and *cake growth*. It was not possible to observe any significant particle penetration into the substrate capillaries. In fact, capillary plugging was finished only a few minutes of filtration, and the mass deposited during these few minutes was very low ($\approx$ 0.1 mg). The cakes consisted of dense regions of particles (agglomerates), separated by chains of particles. The deposit were characterized for their mass, capillary size, and porosity.

Both cake capillary size and porosity were constant with the Pe number in the studied range (0.3 - 0.5), as differences were most likely within experimental reproduceability. The cake capillary size calculated using the permeability method was approximately independent (160 - 227 nm) on the deposited mass in the investigated range (0.7 - 8.7 mg). The capillary size determined from SEM was much larger ($>$ 1000 nm) than what was found from permeability measurements, most likely because the porosity was highest at the top layer of the cake, which was seen in the SEM.

Three methods were used in the determination of the porosity: SEM, the permeability method, and cylindrical capillary model. The porosity calculated with the cylindrical capillary model was $\approx$ 0.97 regardless of the deposited mass. This corresponded well with the porosity calculated from SEM (0.94 - 0.97). In contrast, the porosity calculated from the permeability method decreased with increasing mass. At low deposited mass ($\approx$ 0.7 mg), the porosity calculated from the permeability method was close to 1 (0.999), probably because of the non uniform flow-field through the cake. However, other factors such as surface roughness may have played a role for thin cakes. Smaller porosities (0.989) were obtained when the flow-field was corrected. However, they were still far from the ones calculated from SEM. At high deposited mass ($\approx$ 8 mg), the porosity calculated from the permeability method (0.97) corresponded well with the porosity calculated from the cylindrical capillary model (0.97).

Therefore, using the permeability method, valuable information on the morphology of particle deposits can be obtained. At low deposited mass ($<$ 8 mg), only the determination of the cake capillary size should be done. However, it is still possible to use the cylindrical capillary model to determine the porosity even for low deposited mass, if the primary particle diameter calculated from e.g. the BET surface area, is known.

Chapter 4

Thermal and mechanical stability of nanoparticle deposits

4.1 Introduction

The mechanical and thermal stability of particle deposits is an important issue, since any industrial application requires, that the cakes have a reasonable mechanical strength, as well as a stability toward elevated temperatures. If either is lacking, this can lead to a changing morphology with time, which can be catastrophic for several applications, e.g. in gas-sensors [65].

The as-deposited cake particles are bound together by Van der Waals forces, and as a consequence they are mechanically weak and deteriorate in liquids [22]. In order to stabilize the films, stronger bonding is required between the particles constituting the cake [11]. This is done by a heating process called *sintering*. Sintering has been studied extensively for the manufacturing of composite materials [66, 67, 68, 69, 70]. However, in these studies the initial porosity is typically much lower than that which is obtained for cakes of directly deposited nanoparticles.

It is known, that packing (and thereby also porosity) significantly influences the sinterability of the particles [30] and the existence of macrocapillaries are known to hinder sintering [71].

Several issues are unknown for the sintering of highly porous cakes. In particular, the morphological evolution with the sintering- temperature and time is important, as any change in this will also affect the end use of the cake. Furthermore, it is known [22, 23], that the cake can dislodge from the substrate (delayering) and that cracks may form due to high thermal stresses.

In this chapter, the basic concepts of sintering is first presented. Then, using the permeability method and SEM images, a study is reported in order

to characterize the:

- morphological time and temperature evolution,
- maximum sintering temperature and
- the increase in mechanical stability

As it has previously been found, that the as-deposited, as well as sintered cakes completely disintegrate in liquids [22], a less destructive method for the characterization of the mechanical stability is required. By controlling the condensation of a vapour near its saturation pressure inside the porous cake, one can examine the point (S, relative saturation) at which the cake collapses. An apparatus is presented, in which the deposited cake is placed and subjected to a constant vapour pressure of a condensable species (here: cyclohexane). Subsequent measurement of the permeability is carried out, and it is revealed whether or not the morphology has changed.

4.2 Theory of particle and cake sintering

4.2.1 Sintering kinetics

The transport of mass between particles during sintering, may take place due to [27]:

- viscous flow,
- evaporation and condensation,
- lattice diffusion,
- surface diffusion, and
- grain boundary diffusion

The diffusion mechanisms are all thermally activated, and follow an Arrhenius type expression for their diffusion coefficients [72]. Schaper et al. [73] found sintering of γ-alumina to proceed through the surface diffusion mechanism. Johannessen et al. [27] found the expression for the characteristic time of sintering (τ_f, time required for full coalescence between two identical spheres), for flame-made alumina to be:

$$\tau_f = 2.70 \cdot 10^{23} d_p^4 \exp\left(\frac{8178.49}{T}\right) T \tag{4.1}$$

Equation 4.1 is only valid for the sintering of nanoparticles in flames, however the trend of τ_f on d_p is generally the same: particles tend to grow to a certain size before the sintering kinetics become too slow.

4.2.2 Regimes of sintering of powders

In general, there exists 3 sintering regimes [66, 74], which all occur at increasing temperatures:

1. Initial stage sintering: Particles form sintering necks. The porosity remains approximately constant throughout this stage, as the total capillary volume does not change significantly. Therefore, no shrinkage occurs.

2. Intermediate stage sintering: The particles start sintering more rapidly, forming a solid-phase, with a continuous network of inter-connected capillaries in between. The change in porosity is large and a significant shrinkage occurs.

3. Final stage sintering: The capillarities are closed off and the remaining porosity is removed. The theoretical bulk density is approached.

The investigated temperature range is in this work from 500 °C to 1100 °C. As a rule of thumb, the sintering temperature at which significant densification occurs (intermediate stage sintering), is 2/3 of the melting point temperature of the bulk material [75]. For nanocrystalline materials, this temperature may be even lower, however normally not below 1/2 of the melting point temperature [70, 76]. Alumina has a melting point temperature of 2054 °C, meaning that the *minimum* expected temperature at which densification takes place is approximately 1000 °C. Therefore, the investigated temperature range covers both initial and intermediate stage sintering.

4.2.3 Constrained sintering

When a cake well anchored to a substrate surface is sintered, the cake can only shrink in the direction normal to the substrate surface [66]. This is known as constrained sintering [77]. Due to the one-dimensional shrinkage of the intermediate stage sintering, cracks may form as a result of stress building up [68, 78]. According to Scherer and Garino [78], the maximum stress in the cake during constrained sintering is proportional to the initial density (ρ_0):

$$\sigma_{\max} \propto \left(\frac{\rho_s}{\rho_0}\right)^{\frac{1}{3}} \tag{4.2}$$

where ρ_s is the density of the solid constituting the cake. Equation 4.2 may be written in terms of porosity, since $\rho_0 = (1-\epsilon)\rho_s$:

$$\sigma_{\max} \propto \left(\frac{1}{1-\epsilon}\right)^{\frac{1}{3}} \tag{4.3}$$

Equation 4.3 predicts that the sintering stress becomes higher the more porous the material is, prior to sintering. If these stresses become too high, the cake may crack and separate into "islands" which can continue to sinter free of constraint [78]. Note that, if the cake is not well anchored to the substrate and shrinkage takes place, the cake will shrink in freely in all directions.

4.2.4 Intra- and inter-agglomerate capillaries

Figure 4.1 shows a typical SEM picture of an as-deposited cake (in chapter 5 the morphology shall be discussed more in detail). The cake consists of agglomerated particles spread apart forming large capillaries. These are the so-called inter-agglomerate capillaries (or macro-capillaries) [30] making up for most of the porosity. However, the agglomerates themselves also contain capillaries, which are the intra-agglomerate capillaries (or micro-capillaries) as shown on figure 4.1.

4.2.5 Capillary growth

It is well known [30, 74, 79], that there exists a critical capillary-size above which capillaries will grow, and below which capillaries will shrink. Mayo et al. [74] determined the ratio of the critical capillary-size to the crystallite size to be 1.5:1. Assuming the crystallite size to be equal to the primary particle size ($\approx$ 8 nm), the critical capillary size is found to be 12 nm. The actual capillary size of the micro-capillaries is not known, however an estimate can be made by assuming a simple arrangement of 4 particles in a cubic lattice, such as the one shown in figure 4.2.

If it is assumed, that the distance x (marked on figure 4.2) between the particles determine the capillary size, such as it is indicated on figure 4.2, then the capillary size becomes:

$$d_{\mathrm{c,micro}} = d_p \left(\sqrt{2} - 1\right) = 3.31 \text{ nm} \tag{4.4}$$

This is only a rough estimate. However, it is clear that there will exist capillaries in the cake within this size-range. Given the right kinetics, these capillaries will disappear and the larger will in principle grow. However, the capillary volume gained to the macro-capillaries by the removal of the micro-capillaries can be neglected, since this volume is much too small to contribute to a growth of the macro-capillaries. Therefore, only the macro-capillaries are considered in the following. If the porosity remains constant throughout sintering, which is typical for the initial stage sintering, the capillary-volume also remains constant. Since the capillaries grow, the number of them must decrease to uphold this balance. Assuming parallel, straight and cylindrical capillaries:

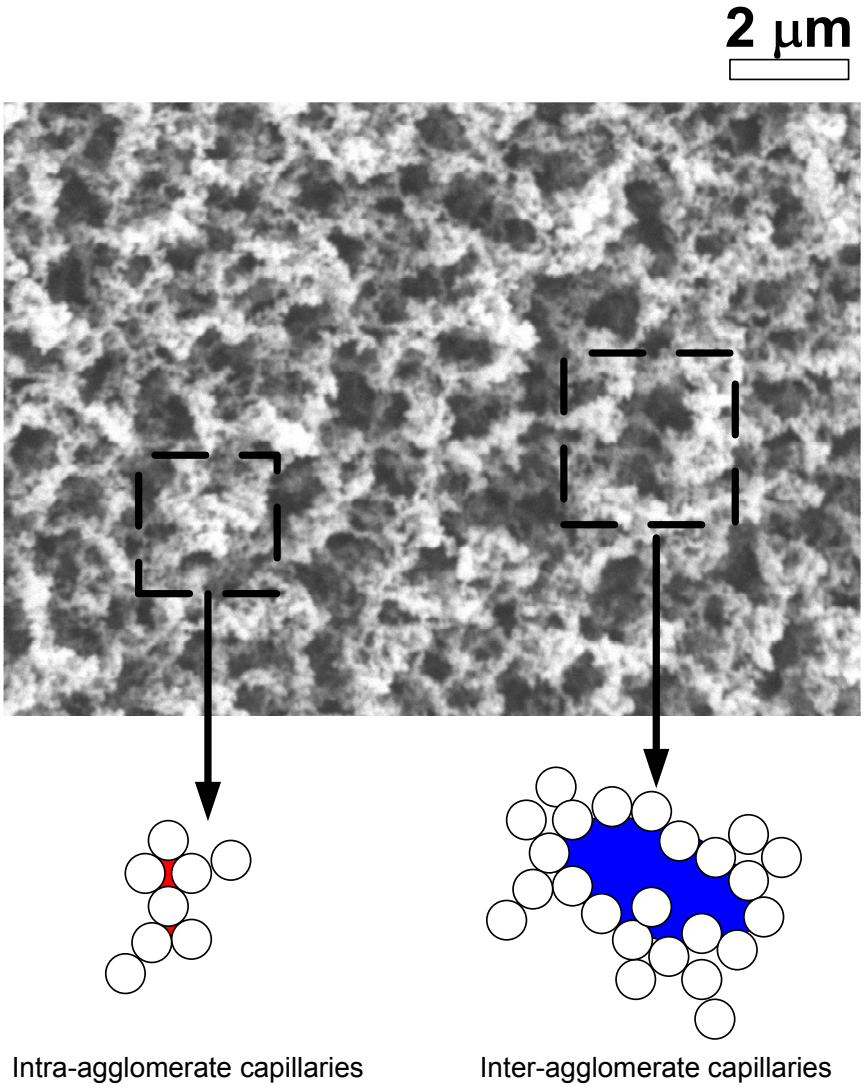


Figure 4.1 SEM of a typical alumina cake deposited on a porous α-alumina substrate. The cake is seen to consist of agglomerated structures and is expected to have a bi-modal capillary size distribution, with the smallest capillaries being between the agglomerate particles and the largest capillaries being between the agglomerates themselves.

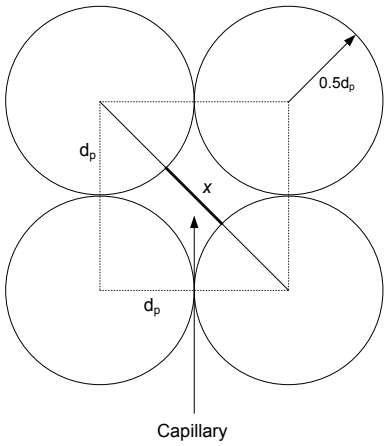


Figure 4.2 The smallest capillaries are formed in the interstices between the intra-agglomerate particles. Here it is assumed that they pack by a cubic arrangement. The resulting capillary size x can then be calculated by simple geometry.

$$\epsilon_c = \frac{n_{c,1}\frac{\pi}{4}d_{cc,1}^2\delta}{V_t} = \frac{n_{c,2}\frac{\pi}{4}d_{cc,2}\delta}{V_t} \tag{4.5}$$

where V_t is the total volume (constant), and $n_{c,x}$, $d_{cc,x}$ is the number- and diameter- of macro-capillaries before ($x = 1$) and after ($x = 2$) sintering respectively. Rearranging for the ratio of capillaries before and after sintering yields:

$$\frac{n_{c,1}}{n_{c,2}} = \left(\frac{d_{cc,2}}{d_{cc,1}}\right)^2 \tag{4.6}$$

4.2.6 Mechanisms for failure

During heating and cooling in the sintering process, cracks and delayering of the structure has been observed to occur for cakes of alumina and magnesia deposited on $\alpha-$alumina tubes by the filtration method [22, 23]. Some of the mechanisms leading to cracking and delayering are:

Thermal expansion mismatch

If the substrate and the cake has different thermal expansion coefficients (α_{TEC}), sheer stresses build up at the interface between the two. The stress by thermal mismatch (σ_{TEC}) for two layers attached to each other with different TEC and heated by ΔT is:

$$\sigma_{\mathrm{TEC}} = E_1 \left(\alpha_{\mathrm{TEC},1} - \alpha_{\mathrm{TEC},2}\right) \Delta T \tag{4.7}$$

where E_1 is the Young modulus of material 1, assumed to have a higher thermal expansion coefficient α_1 than material 2 α_2. For a cake deposited on the inside of a substrate tube, this leads to the formation of both *axial* and *radial* cracks, as shown on figure 4.3. Since the cake will be more elastic than the substrate due to the high porosity and the van der Waals bonding between the constituent particles [80], the cracks are likely to occur during cooling when the cake has become more rigid after the formation of sinter-necks.

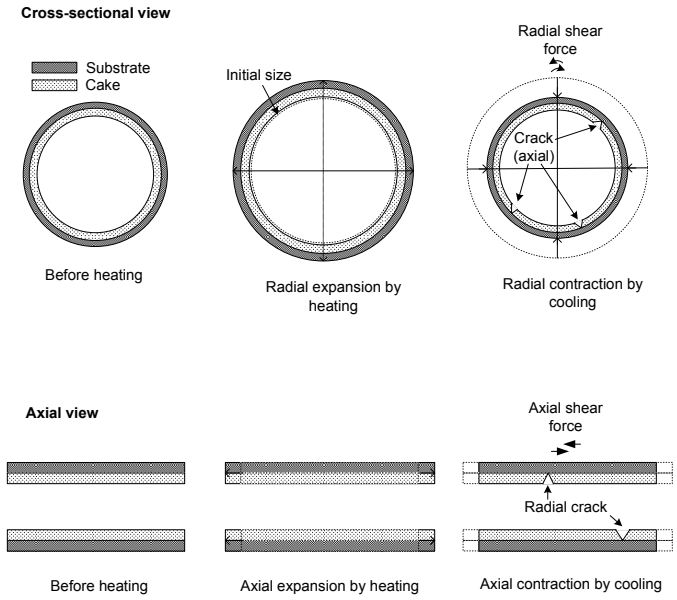


Figure 4.3 Example of the thermal expansion mismatch during sintering/heating of cakes deposited on the inside of a porous substrate. The thickness of the cake has been scaled up in order to show qualitatively the formation of cracks. Cracks occur due to the radial and axial shear forces which can be calculated from equation 4.7.

Constrained shrinkage

If the temperature becomes high enough for particle growth and densification to occur, then shrinkage will also take place (see section 4.2.2). As mentioned in section 4.2.3, the substrate constrains the shrinkage to the dimension normal to the surface if the cake is well anchored to it. For cakes deposited on the inside of a porous substrate tube, this means that the layer can only shrink radially, which leads to the buildup of stresses given by equation 4.3. Most likely, this is the reason why magnesium oxide (MgO) cakes

deposited on α-alumina substrates by the filtration method form cracks during sintering, as these cakes shrink significantly during sintering [23]. One can further not rule out the combination of this mechanism with the thermal expansion mismatch, since the α_{TEC} for MgO is $13.5 \cdot 10^{-6}$ °C^{-1} while Al_2O_3 has an α_{TEC} of $8 \cdot 10^{-6}$ °C^{-1}.

Pinhole defects at substrate capillaries

The porous substrate consists of sintered grains of several microns in size. During the cake sintering, these grains will expand according to their TEC. This expansion is likely to compress the plug of particles in between the capillary mouth. After sintering, when the cake has become more rigid due to the formation of sintering necks, the grains contract again pulling the sintered plug with them. As a result, pinhole defects just above the capillary mouth may form, as shown on figure 4.4.

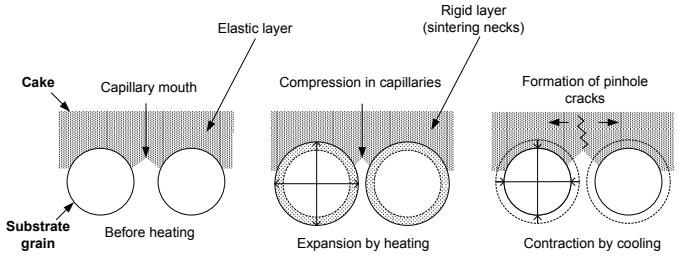


Figure 4.4 Pinhole defects occurring due to the expansion (compression of plug) and contraction in the substrate capillaries during sintering.

4.3 Experimental

4.3.1 Sintering study

Porous alumina cakes were deposited on the porous α-alumina substrates for 10 (thin layers, approximately 3 μm[1]) and 60 min (thick layers, approximately 28 μm) using the standard flame- and precursor conditions (cf. table 2.4). After deposition they were characterized by the permeability method. The cakes were then sintered in an oven at various temperatures ranging from 500°C to 1100°C for 2 or 10 hours, using a heating ramp of 20°C/min, and a cooling ramp of 5°C/min. Permeability measurements and SEM were carried out subsequently to investigate the effects sintering. The experiment labels, estimated thickness of the deposits, along with the sintering temperatures and times are shown in table 4.1 for thin layers and table 4.2 for thick

[1]The thickness was estimated from the filtration curves and assuming a constant porosity of 0.95.

layers. The Pe number was approximately the same for all depositions (0.1 - 0.2).

Table 4.1 Experimental overview for sintering of thin layers at various times. All substrates were 5.8 cm long. Pressure drop in all depositions was $\Delta P = 300$ mbar (Pe ≈ 0.3). All deposition times: 10 minutes.

Label	Deposited mass	Estimated thickness[1]	Sintering time	Sintering temperature
-	mg	μm	hours	°C
tde-09	0.7	3.0	10	1100
tde-10	0.74	3.2	10	900
tde-11	0.8	3.5	10	700
tde-12	0.74	3.2	10	500
tde-13	0.77	3.3	2	1100
tde-15	0.76	3.3	2	900
tde-16	0.76	3.3	2	700
tde-17	0.80	3.5	2	500

[1]: Estimated using a constant porosity of 0.95.

Table 4.2 Experimental overview for sintering of thick layers at various times. All substrates were 5.8 cm long. Pressure drop in all depositions was $\Delta P = 500$ mbar (Pe ≈ 0.5). All deposition times: 1 hour.

Label	Deposited mass	Estimated thickness[1]	Sintering time	Sintering temperature
-	mg	μm	hours	°C
tde-22a[2]	6.5	28.0	2	1100
tde-22b[2]	6.5	28.0	2	900
tde-23a[3]	6.3	27.3	2	700
tde-23b[3]	6.3	27.3	2	500

[1]: Estimated using a constant porosity of 0.95.

[2]: Cakes tde-22a and tde-22b were deposited on the same substrate. After deposition they were cut in two halves and sintered.

[3]: The same applied for tde-23a and tde-23b.

4.3.2 Mechanical stability study

Figure 4.5 shows a schematic of the apparatus. Two bubble flasks were filled with cyclohexane and placed in an oven at a constant temperature (30°C). Two streams of nitrogen were saturated by passing them through the flasks. Subsequent mixing with dry streams of nitrogen allowed the control of the relative saturation (S) defined as:

$$S = \frac{P_c}{P_{c,sat}(T)} \tag{4.8}$$

where P_c is the partial pressure of cyclohexane and $P_{c,sat}(T)$ is the saturation pressure of cyclohexane at temperature T.

All flows were controlled by mass-flow controllers to allow as stable a flow as possible. After mixing, the two streams were led to a cell containing the substrate and cake. They were passed on the outside and inside of the cake and substrate in order to assure a homogeneous concentration profile of the cyclohexane. After exposing the substrate and cake for approximately 30 minutes, the streams were cut off, and the substrate and cake was dried. During this time, the amount of cyclohexane evaporated from the saturators, did not sufficiently decrease the total volume of cyclohexane. This assured, that the gas leaving the saturators was infact at all times completely saturated. Subsequently permeability measurements were carried out and the morphology of the cake was characterized.

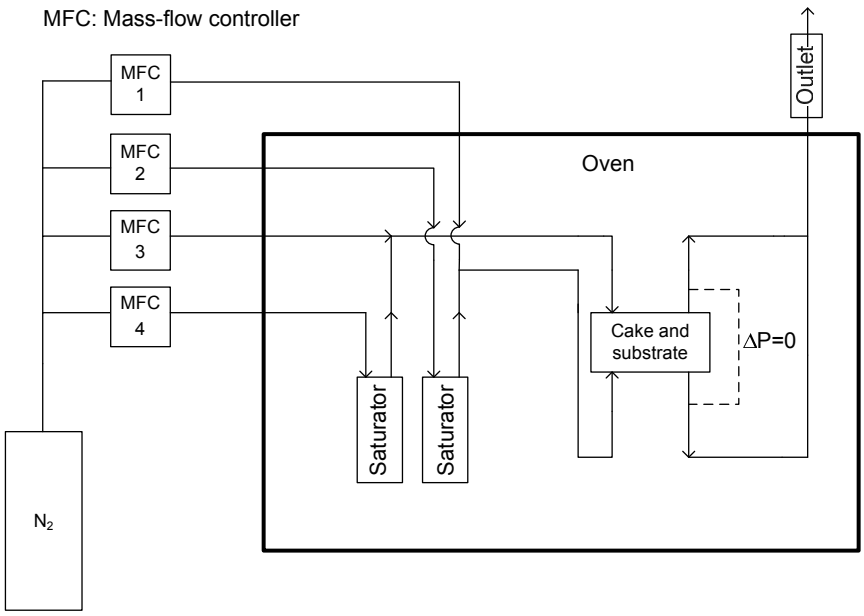


Figure 4.5 Apparatus for the investigation of the mechanical stability of deposited cakes, by controlled condensation of cyclohexane. Four mass-flow controllers (MFC) deliver streams of dry nitrogen. Two of the streams are passed through the two saturator bottles, and is subsequently mixed with the dry nitrogen to control the relative saturation (S). These streams are fed to the substrate and cake on both sides.

The forces acting in the cake are the capillary-forces which are present due to the condensation of cyclohexane within the porous structure. As it

will be shown in section 4.7.1, these forces can be several orders of magnitude larger than the Van der Waals forces which keep the cake constituent particles together. With this method, one can examine not only the stability of the cakes in a near condensing environment, but also investigate indirectly whether or not sinter necks actually form, as such should greatly increase the mechanical stability of the cakes under these conditions.

The cakes which were tested with this method was deposited using the conditions shown in table 4.3.

Table 4.3 Deposition conditions and estimated cake thickness for the cakes to be tested for their mechanical stability. Pressure drop during all depositions was $\Delta P = 500$ mbar. All deposition times: 1 hour.

Label	Deposited mass	Estimated thickness	Sintering time	Sintering temperature
-	mg	μm	hours	°C
tde-29	8.19	35.5	-	-
tde-30	5.75	24.9	-	-
tde-31	8.66	37.5	2	700

4.4 Results of the sintering study

4.4.1 Permeability curves

Figure 4.6 shows the permeability, before deposition, after deposition, and after sintering for the thin (3 μm) cakes at various sintering temperatures (2 hours sintering).

The permeability of thin cakes sintered at 10 hours and thick cakes sintered at 2 hours, is not shown here, since the trend is the same: The permeability generally increases with increasing sintering temperature. At 1100°C, the permeability had increased to more than that of the clean substrate, thereby indicating that the substrate itself had become more permeable due to capillary growth in the substrate. The permeability analysis can therefore not be carried out for cakes sintered at this temperature, as it is a requirement of the method that the substrate morphology stays constant after the initial measurement.

4.4.2 Sintering of thin cakes - time dependency

Figure 4.7 shows the growth of the measured cake capillary size for thin cakes of approximately 3 μm thickness (cf. table 4.1) for 2 and 10 hours of sintering. The growth is shown as the ratio of the cake capillary size after sintering to the capillary size of the as-deposited cake. A trend is seen, in which the cake capillary size increases for increasing sintering temperatures.

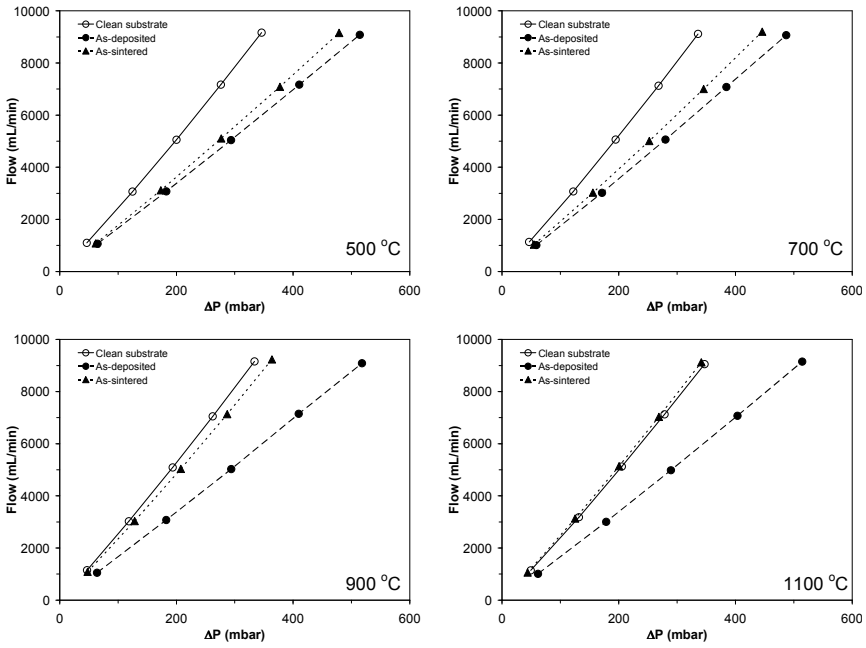


Figure 4.6 Permeability curves before deposition, after deposition and after sintering of thin cakes (3 μm, alumina) on a α-alumina substrate at various sintering temperatures. Holding time: 2 hours.

Increasing the time of sintering also increases the growth. The values for cakes sintered at 1100°C (tde-09 and tde-13) have been omitted.

Figure 4.8 shows a SEM image for the cake sintered at 700°C, which is the temperature at which the largest difference between the 2 hour sintering and the 10 hour sintering is observed. The figure reveals, that micro-cracks form at this temperature.

4.4.3 Sintering of thick versus thin cakes

Figure 4.9 shows the ratio of the cake capillary size after sintering to the capillary size of the as-deposited cakes for thick (28 μm), and thin (3 μm) cakes sintered for 2 hours at various temperatures. As for the thin cakes, the thick cakes also grow in cake capillary size after sintering, with increasing growth at higher temperatures. The thick cakes however seem to grow less in capillary size at increasing temperatures than that observed for the thinner ones.

Figure 4.10 shows a SEM image for the thick cake sintered at 900°C. This is the temperature at which the largest difference in growth of the cake capillary size between the 2 hour sintered cakes (thick versus thin) is seen. The figure reveals extensive micro-cracks forming, however the cake is still overall intact. The small black spots are most likely due to residue during the sample preparation in SEM, which requires breaking of the substrates.

Figure 4.11 shows a SEM image for the thick cake sintered at 1100°C. The cake is completely cracked and torn apart, which was also evident from permeability measurements.

4.5 Discussion

4.5.1 Effect of holding time

As it would seem from figure 4.7 the effect of increasing the holding time, becomes significant for sintering temperatures above 700 °C. One may be tempted to conclude that the particles have grown more due to slow kinetics, thereby increasing the capillary size according to the cylindrical capillary model. However, no significant particle growth due to sintering takes place below 900°C [81, 82], even for holding times as long as 15 hours [83]. The reason for the difference in capillary growth can therefore not be explained by the kinetics.

Since SEM showed micro-cracks in the thin cakes sintered for 2 hours at 700°C (cf. figure 4.8), a more likely explanation is that the effect of micro-cracking may increase at even longer holding times. This is in particular the case for thin cakes, when cracks occur near the interface between the substrate and cake. However, SEM images should be captured in order to

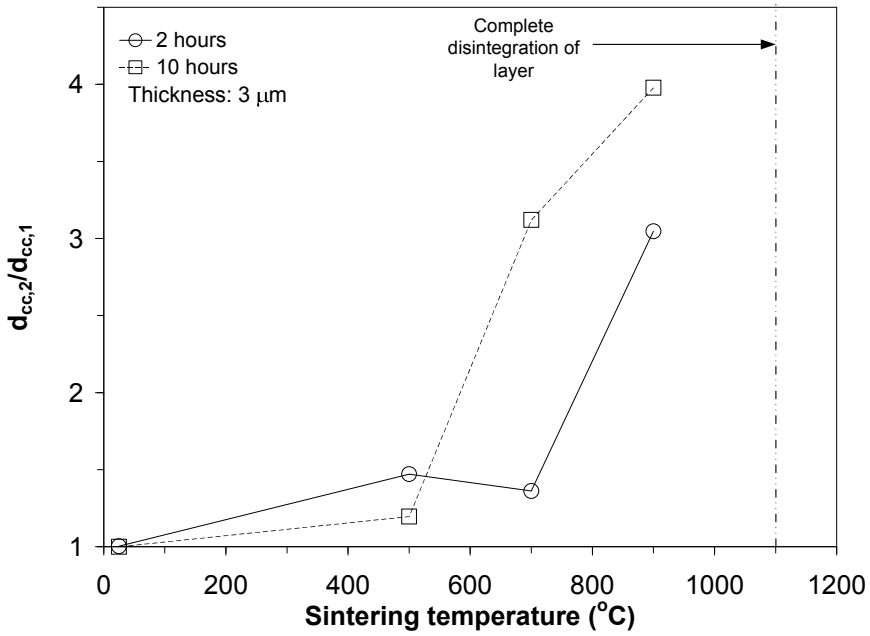


Figure 4.7 Ratio of the cake capillary size after sintering ($d_{cc,2}$) to the cake capillary size before sintering ($d_{cc,1}$) for thin cakes (3 μm) sintered for 2 and 10 hours at various temperatures.

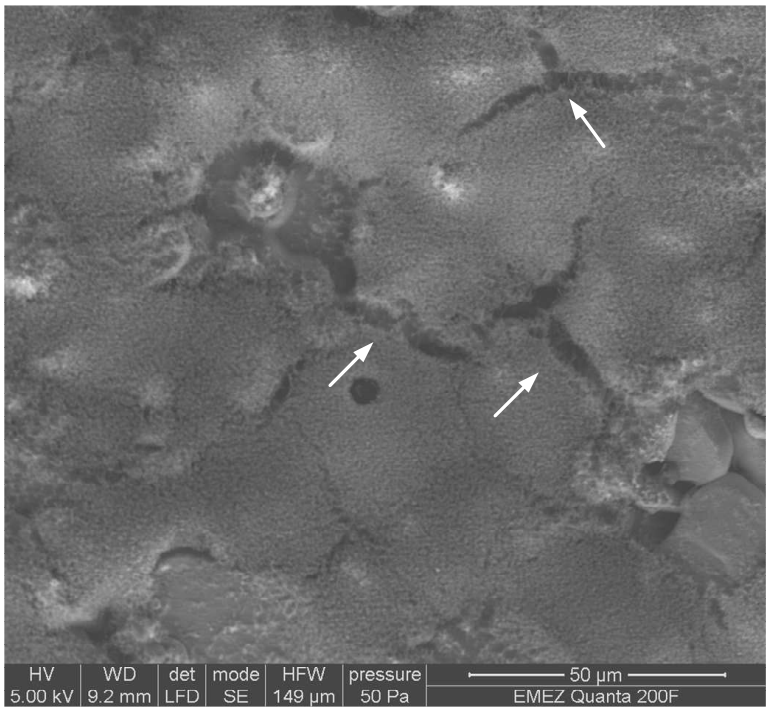


Figure 4.8 Thin cake (3 μm) sintered at 700°C for 2 hours. Experiment label: tde-16 (see table 4.1). Several micro-cracks (white arrows) are observed.

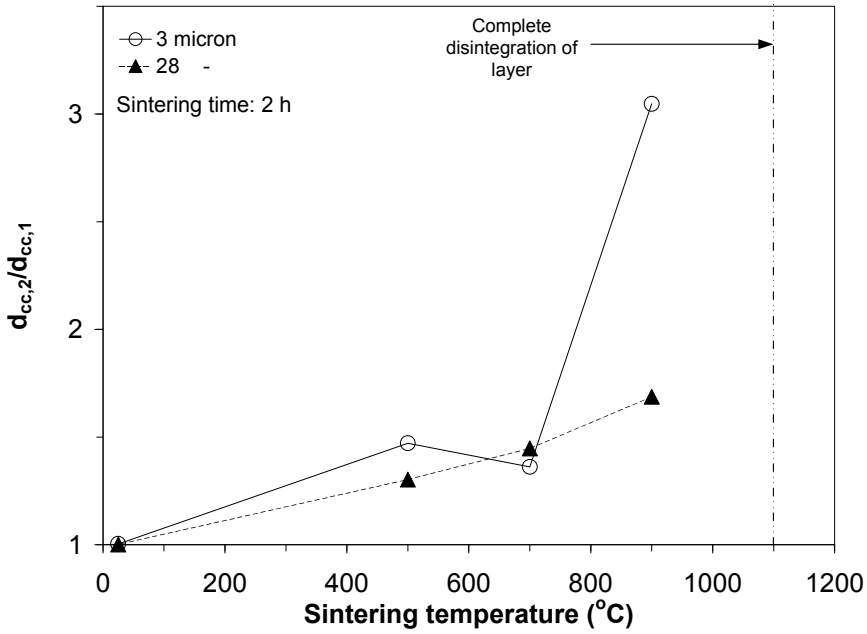


Figure 4.9 Ratio of the cake capillary size after sintering ($d_{cc,2}$) to the cake capillary size before sintering ($d_{cc,1}$) for thick (28 μm) and thin cakes (3 μm) sintered for 2 hours at various temperatures.

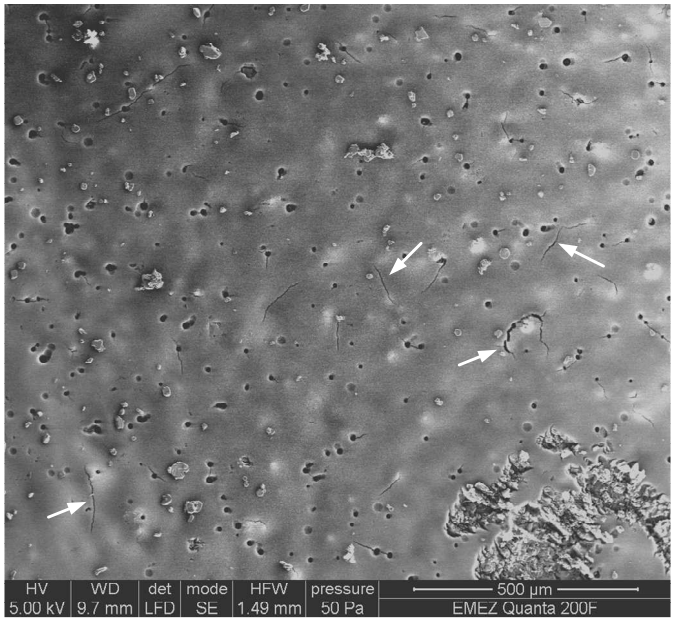


Figure 4.10 Thick cake (28 μm) sintered at 900°C for 2 hours. Experiment label: tde-22b (see table 4.2). Significant micro-cracking is observed (white arrows), however the cake is still overall intact.

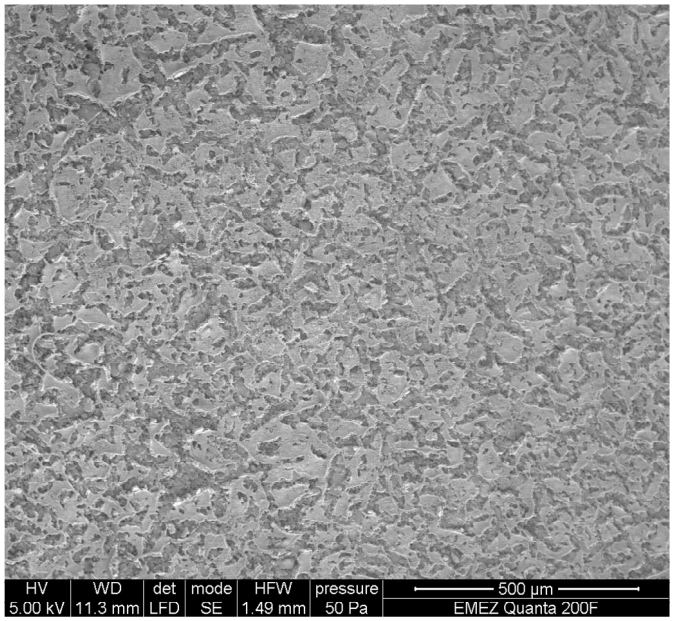


Figure 4.11 Thick cake (28 μm) sintered at 1100°C for 2 hours. Experiment label: tde-22a (see table 4.2). The cake is completely cracked and torn apart at this temperature.

verify this.

4.5.2 Effect of cake thickness

Comparing the thin cakes to the thick cakes on figure 4.9, it can be seen that the thick cakes increase less in capillary size with increasing temperature. This indicates, that the effect of micro-cracking becomes less important in the permeability measurements for thick cakes, as most of the flow resistance is still in through the cake.

4.5.3 Capillary growth - micro cracking

Regardless of the thickness and holding time, a growth in the average cake capillary size is seen. This growth is too large to be explained by the loss of capillaries using equation 4.5. As an example, 54% of the capillaries would have to be removed to make up for the increased capillary size when sintering the thin cake for 2 hours at 500°C:

$$\frac{n_{c,1} - n_{c,2}}{n_{c,1}} = 1 - \left(\frac{d_{cc,2}}{d_{cc,1}}\right)^{-2} = 1 - (1.47)^{-2} = 0.54$$

From figure 4.8 and 4.10 it is more likely that the growth in cake capillary size is due to the formation of micro-cracks. These micro-cracks have no apparent direction and do not extend for long distances. Therefore, it is unlikely that the cracks have formed due to TEC mismatch (cf. section 4.2.6). A possible mechanism is that cracks form due to the expansion and contraction of the substrate grains near the substrate capillaries, as discussed in section 4.2.6.

4.5.4 Micro-cracking - effect on permeability

A simple semi-quantitative analysis was carried out in order to estimate the effect of the cracks on the permeability. First, a number of assumptions were made (subscript 1 and 2 denote before and after sintering, respectively):

1. The capillary size distribution after sintering is bi-modal due to the formation of cracks. The capillary size of the cake ($d_{cc,1}$) does not change.

2. The number of substrate capillaries ($n_{c,s}$) is determined assuming a substrate porosity of 30% and cylindrical substrate capillaries with d_{cs} found from the permeability measurements for the clean substrate.

3. The number of cake capillaries prior to sintering ($n_{c,1}$) is determined from the initial measured cake capillary size ($d_{cc,1}$), assuming cylindrical cake capillaries and a cake porosity of $\epsilon_c = 95\%$.

4. The number of cracks in the cake per unit area after sintering ($N_{cr,2} = \frac{n_{cr,2}}{A_c}$) can be estimated from SEM images. The total number of cracks ($n_{cr,2}$) is then found by multiplying with the total area A_c.

5. The porosity is constant during and after sintering. This assumption is reasonable for initial stage sintering, i.e. temperatures below 1000°C.

6. The cracks are modelled as cylindrical and straight capillaries.

7. The flow through each crack (q_{cr}) and cake capillary (q_c) is found using Poiseuille's law.

8. The total flow is found as the sum of the flow through the cake capillaries and the crack capillaries.

The calculations are done for the thin cake sintered at 700°C (tde-16).

The flow through a single crack and cake capillary is estimated from Poiseuille's law, which for the cake capillaries before sintering:

$$q_{c,1} = \frac{\pi d_{cc,1}^4}{128\mu_g}\frac{\Delta P_c}{\delta_c} \tag{4.9}$$

where ΔP_c is the pressure-drop across the cake equal to $P_1 - P_i$. The value of P_i (interface pressure between cake and substrate) depends on the substrate morphology, and in section 3.2.4 it was assumed that this quantity could be expressed by a linear expansion in the total pressure drop ΔP with expansion coefficient α_j ($j = (1, 2)$). Therefore $\Delta P_c = P_1 - P_i = P_0 + \Delta P - P_0 - \alpha_j \Delta P = (1 - \alpha_j)\Delta P$, and equation 4.9 can be written in terms of the total pressure-drop:

$$q_{c,1} = \frac{(1-\alpha_1)\pi d_{cc,1}^4}{128\mu_g}\frac{\Delta P}{\delta_c} \tag{4.10}$$

Similarly, after sintering ($d_{cc,2} = d_{cc,1}$, since the cake capillaries are assumed not to grow):

$$q_{c,2} = \frac{(1-\alpha_2)\pi d_{cc,1}^4}{128\mu_g}\frac{\Delta P}{\delta_c} \tag{4.11}$$

The cracks form only after sintering, and the corresponding equation for flow through each crack capillary is:

$$q_{cr,2} = \frac{(1-\alpha_2)\pi d_{cr,2}^4}{128\mu_g}\frac{\Delta P}{\delta_c} \tag{4.12}$$

The total flow ($Q_{t,j}$) before ($j = 1$) and after ($j = 2$) sintering is then found by:

$$Q_{t,j} = n_{c,j} q_{c,j} + n_{cr,j} q_{cr,j} \tag{4.13}$$

Before sintering, since no cracks are present ($n_{cr,1} = 0$), the total flow is:

$$Q_{t,1} = n_{c,1} q_{c,1} = \frac{(1-\alpha_1)\pi d_{cc,1}^4}{128\mu_g} \frac{\Delta P}{\delta_c} \tag{4.14}$$

Similarly, after sintering:

$$Q_{t,2} = \frac{(1-\alpha_2)\pi \Delta P}{128\mu_g \delta_c} \left(n_{c,2} d_{cc,1}^4 + n_{cr,2} d_{cr,2}^4 \right) \tag{4.15}$$

The ratio of the flow after sintering to the flow before sintering is known from permeability measurements, therefore by dividing equation 4.15 with 4.14 this can be expressed as:

$$\frac{Q_{t,2}}{Q_{t,1}} = \frac{1-\alpha_2}{1-\alpha_1} \left(\frac{n_{cr,2}}{n_{c,1}} \left(\frac{d_{cr,2}}{d_{cc,1}} \right)^4 + \frac{n_{c,2}}{n_{c,1}} \right) \tag{4.16}$$

The number of cake capillaries is found from a balance on the total capillary volume:

$$n_{c,1} \frac{\pi}{4} d_{cc,1}^2 \delta_c = A_c \delta_c \epsilon_c \Rightarrow$$

$$n_{c,1} = \frac{4 A_c \epsilon_c}{\pi d_{cc,1}^2} \tag{4.17}$$

The ratio of the number of cake capillaries after sintering ($n_{c,2}$) to the number of cake capillaries before sintering ($n_{c,1}$) is also found from a volume balance, since the porosity during sintering is approximately constant:

$$n_{c,1} \frac{\pi}{4} d_{cc,1}^2 \delta_c = n_{c,2} \frac{\pi}{4} d_{cc,1}^2 \delta_c + n_{cr,2} \frac{\pi}{4} d_{cr,2}^2 \delta_c \Rightarrow$$

$$\frac{n_{c,2}}{n_{c,1}} = 1 - \frac{n_{cr,2}}{n_{c,1}} \left(\frac{d_{cr,2}}{d_{cc,1}} \right)^2 \tag{4.18}$$

The ratio of the number of crack capillaries ($n_{c,c}$) to the number of initial cake capillaries ($n_{c,1}$) can be found, since the number of crack capillaries per area ($N_{cr,2}$) is known (counted from SEM):

$$\frac{n_{cr,2}}{n_{c,1}} = \frac{N_{cr,2} A_c \pi d_{cc,1}^2}{4 A_c \epsilon_c} = \frac{N_{cr,2} \pi d_{cc,1}^2}{4 \epsilon_c} \tag{4.19}$$

Inserting equation 4.18 into equation 4.16 yields:

$$\frac{Q_{t,2}}{Q_{t,1}} = \frac{1-\alpha_2}{1-\alpha_1} \left(\frac{n_{cr,2}}{n_{c,1}} \left(\frac{d_{cr,2}}{d_{cc,1}} \right)^4 + 1 - \frac{n_{cr,2}}{n_{c,1}} \left(\frac{d_{cr,2}}{d_{cc,1}} \right)^2 \right) \tag{4.20}$$

or simply:

$$\frac{Q_{t,2}}{Q_{t,1}} = \frac{1-\alpha_2}{1-\alpha_1} \left(1 - X \cdot Y^2 + X \cdot Y^4 \right) \tag{4.21}$$

with $X = \frac{n_{cr,2}}{n_{c,1}}$ (eq. 4.19) and $Y = \frac{d_{cr,2}}{d_{cc,1}}$. Since X is known, Y may be estimated by solving 4.21.

The parameters required to solve eq. 4.21 is determined by the permeability method, and are listed in table 4.4.

Table 4.4 Parameters for equation 4.21 determined with the permeability method. Experiment label **tde-16**.

$\frac{Q_{t,2}}{Q_{t,1}}$	$d_{cc,1}$	α_1	α_2
-	nm	-	-
1.060	183	0.703	0.745

The number of cracks for this cake (tde-16) is counted to approximately 3 cracks per 2500 μm^2. The ratio then becomes (eq. 4.19):

$$X = \frac{n_{cr,2}}{n_{c,1}} = \frac{\frac{3}{(50 \cdot 10^{-6})^2} \pi (183 \cdot 10^{-9})^2}{4 \cdot 0.95} = 3.3 \cdot 10^{-5}$$

Solving 4.21 for Y then yields:

$$Y = 9.2 \Rightarrow d_{cr,2} = 1.7 \ \mu\text{m}$$

The calculated size of the cracks is somewhat smaller than observed by SEM. This may be due several weak points in the assumptions, such as cylindrical cracks, but also poor statistics in the counting of cracks may

affect the result. However, the calculation clearly shows, that it is possible to explain the increase in permeability, shown in figure 4.6, as the result of the formation of micron-sized cracks.

The calculation for the cake sintered at 700°C was only an example. As mentioned, the growth of the cake average capillary size was already too large at 500°C to be explained by the removal of capillaries. Sintering experiments carried out for flame-made (FSP) Al_2O_3 powders have shown the formation of hard-agglomerates due to sintering necks [84]. Therefore, if the main mechanism of crack formation is because of an increase in the rigidness of the cake, it is expected that similar cracks form already in cakes sintered at 500°C.

4.5.5 Cake disintegration

Up to 900°C, the formation of cracks were seen, however the overall cake was intact. Most likely, this is because no significant shrinkage takes place. The porosity should therefore remain constant up to this temperature. However, at a sintering temperature of 1100°C, the cakes were completely destroyed. Also the substrate morphology changed at this temperature, since the permeability was higher after sintering than the clean substrate. Figure 4.12 shows a close up of the cake structure after sintering at 1100°C. There are huge gaps in between the particle covered areas, however the structure of areas covered by particles looks similar to that before sintering. This indicates, that the cake has shrunk in all directions, leading to a reduction in both height and area and eventually to its complete disintegration.

Powder studies on the sintering of nano-sized alumina particles [81, 82, 83] suggest, that sintering temperatures above 1000°C brings about a large increase in the grain-size of the alumina, most likely due to the formation of α-alumina from γ-alumina. The size of the particles after sintering is not known, as no BET data is available for this study. As growth requires mass to be transferred to the particles, the growth may be limited, since the coordination number (number of particles touching per particle) will be smaller for higher porosities. Since powders generally pack with high porosities due to their agglomerated structures, data from alumina powder sintering studies may be used to estimate the particle size after sintering in the cake. This was done using the data of Strobel et al. [82], who studied the sintering of flame-made alumina powders. There it was found, that the powder particles grew from approximately 12 nm (specific surface area = 126 m^2/g) to approximately 17 nm (specific surface area = 90 m^2/g) after sintering at 1100°C for 4 hours. Recalling that sintering kinetics are closely connected to the particle size (cf. eq. 4.1), it is likely that this particle size was the maximum obtainable for sintering at this temperature. As the particles formed in this work are approximately 8 nm initially, it is probable that they will grow also to 17 nm, or by a factor of 2.13.

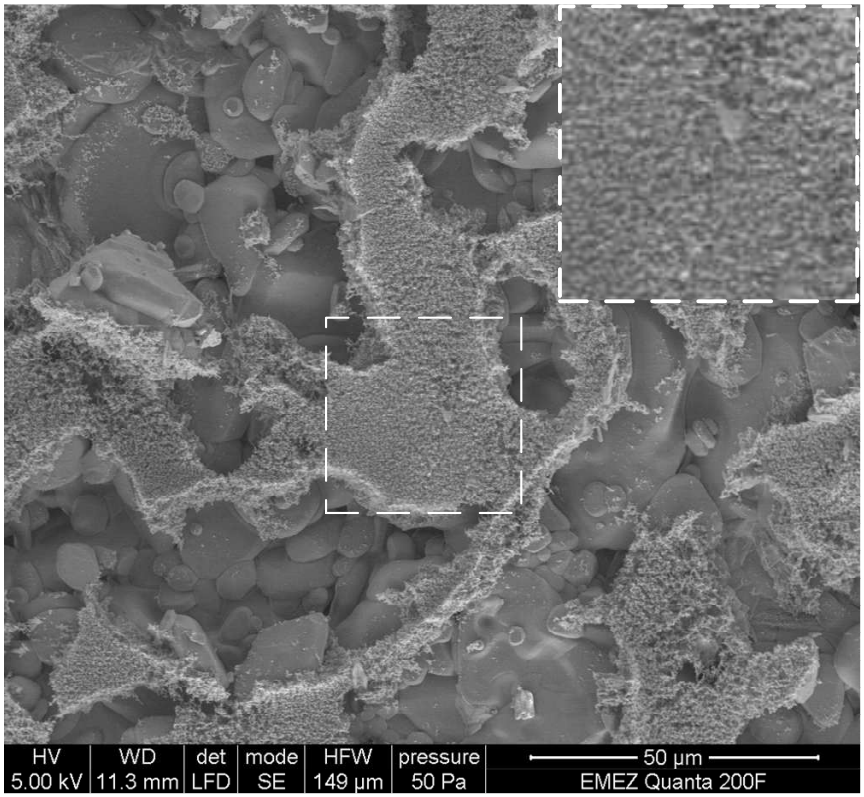


Figure 4.12 Increased zoom of cake sintered at 1100°C. Experiment label: tde-22a (see table 4.2). The micro-structure seems to have been preserved inside the particle covered area.

It was not possible to measure the porosity from the shrinkage, as the thickness could not be determined from SEM. However, as mentioned, the structure of the area covered by particles looked intact after sintering. Therefore, an *estimate* of the shrinkage shall be made using the cylindrical capillary model. Assuming, that the cake capillaries do not grow, the ratio of the porosity before and after sintering inside the particle covered area is estimated from the cylindrical capillary model:

$$\left.\begin{array}{l} \frac{d_{cc}}{d_{p,1}} = \frac{2}{3}\frac{\epsilon_{c,1}}{1-\epsilon_{c,1}} \\ \frac{d_{cc}}{d_{p,2}} = \frac{2}{3}\frac{\epsilon_{c,2}}{1-\epsilon_{c,2}} \end{array}\right\} \Rightarrow \frac{d_{p,1}}{d_{p,2}} = \frac{\epsilon_{c,2}(1-\epsilon_{c,1})}{\epsilon_{c,1}(1-\epsilon_{c,2})} \tag{4.22}$$

where $d_{p,x}$ and ϵ_x is the particle size and cake porosity respectively before ($x = 0$) and after ($x = 1$) sintering. Since the cake porosity is expected to be high even after sintering, the fraction $\frac{\epsilon_{c,2}}{\epsilon_{c,1}}$ will be close to 1, which simplifies eq. 4.22 to:

$$\frac{d_{p,1}}{d_{p,2}} \approx \frac{1-\epsilon_{c,1}}{1-\epsilon_{c,2}} \tag{4.23}$$

If the cake is allowed to shrink in all 3 dimensions (x, y, z), the shrinkage should occur isotropically. A mass-balance on a control volume $\Delta V = \Delta x \Delta y \Delta z$ then yields:

$$(1-\epsilon_{c,1})\Delta V = (1-\epsilon_{c,2}) f_s^3 \Delta V \tag{4.24}$$

where f_s is the fractional shrinkage in all 3 dimensions. Solving for f_s yields:

$$f_s = \left(\frac{1-\epsilon_{c,1}}{1-\epsilon_{c,2}}\right)^{\frac{1}{3}} \tag{4.25}$$

Inserting eq. 4.23 gives:

$$f_s = \left(\frac{d_{p,1}}{d_{p,2}}\right)^{\frac{1}{3}} = \left(\frac{8}{17}\right)^{\frac{1}{3}} = 0.78 \tag{4.26}$$

The area covered by particles shrinks proportional to f_s^2. The area covered by particles should therefore be approximately equal to $f_s^2 = 61\%$ of the initially covered area. Computer programs (OPTIMAS 6.51 and Paint Shop Pro 8, PSP) were used to determine this area from SEM over a total area of 0.91 mm^2, as shown on figure 4.13. The calculated particle covered

area was found to be 62% (OPTIMAS) and 60% (PSP) of the total area, which is in excellent agreement with the one estimated from equation 4.26.

Equation 4.26 was derived from the assumption, that the cake capillary size is kept constant throughout sintering. This may seem like a rough assumption, however up till this temperature, no capillary growth was seen and the permeability increase was found to be described as an effect of crack formation, rather than capillary growth (cf. section 4.5.4). Given the excellent agreement between SEM and equation 4.26, it is likely that the cake has ruptured due to the free shrinkage of it.

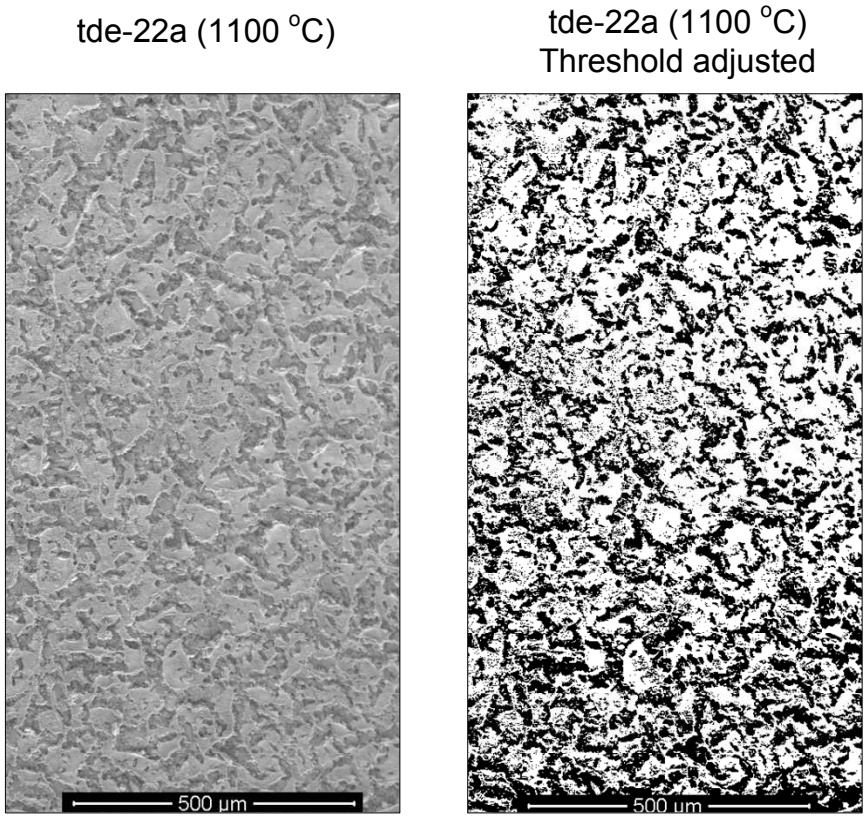


Figure 4.13 SEM of the cake sintered at 1100°C before and after image processing with Paint Shop Pro 8. Experiment label: tde-22a (see table 4.2). The shaded area was calculated to be 60% of the total area in excellent agreement with the assumption of free shrinkage.

4.6 Sintering stability - conclusions

The thermal stability of as-deposited (thin- and thick-) alumina cakes was investigated by sintering the cakes at temperatures from 500 - 1100 °C. Using the permeability method and SEM, the cake morphology was characterized. The cake average capillary size determined by permeability measurements increased more for thin cakes than for thick cakes. This suggested that thin cakes were more fragile than thicker ones. Furthermore, the permeability increased to an extent which could not be explained due to pure capillary growth. From SEM it was observed that cracks were present in cakes sintered at 700°C and 900°C. The cracks were micron sized and had no apparent direction. It is possible, that they formed as a result of the compression-contraction in the substrate capillaries during the heating and cooling. Most likely during cooling, as sinter necks would have formed which made the cake more rigid. A simple model was shown to be able to describe the observed permeability increase due to the formation of micron-sized "cracks", for cakes sintered at 700°C. Cracks were most likely also evident in cakes sintered at 500°C, since the increase in the cake average capillary size was too large to be explained by the removal of capillaries.

The cake porosity up to 900°C was approximately constant, since the cake was intact (apart from a few micro-cracks), while at 1100°C the cake completely disintegrated due to fast coalescence of particles, which led to the free shrinkage of the cake and a decrease in porosity. The assumption of free-shrinkage was backed up by a simple model, which was verified by SEM image analysis using two different computer programs. Using the model, the area covered by particles after sintering was found to be 61% of the total area initially covered. In comparison SEM image analysis found 60 - 62 %.

The results of this study show, that the cakes maintain their overall integrity up to up to 900°C, however with the formation of small micron sized cracks.

4.7 Stability measurements

4.7.1 Unsintered cakes

Figure 4.14a-b shows the permeability of two as-deposited cakes subjected to several degrees of relative saturation (S) of cyclohexane vapours. At $S = 0.7$ (figure 4.14b), no change in the permeability is seen. At $S = 1.0$ the permeability increases drastically for both cakes, indicating a destruction of the structure of the as-deposited cakes. For one of the cakes, a change in the permeability is seen already at $S = 0.9$ (fig. 4.14a). This shows, that the destruction of the cake occurs only at high relative humidities. In fact, it has been shown that low relative saturation should yield an *increase* in the particle-particle adhesion forces due to the formation of a liquid meniscus between the neighbouring particles [85]. The lower the S, the higher the curvature of this meniscus, and the attractive capillary-forces therefore become larger.

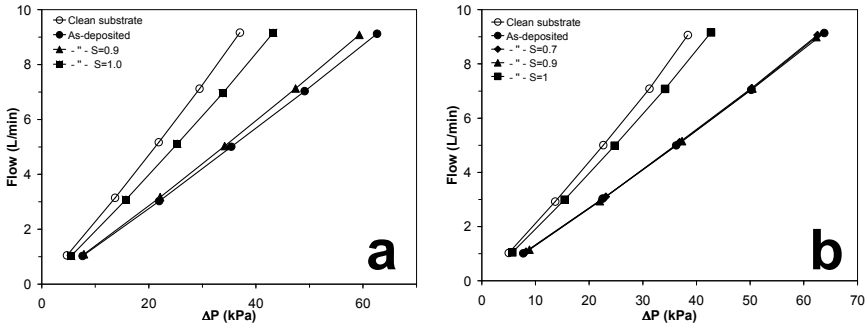


Figure 4.14 Permeability curves for two as deposited cakes (a,b) exposed to several values of relative saturation of cyclohexane (S). Experiment labels: (a): tde-29 and (b): tde-30 (see table 4.3). The permeability is seen to increase when S is near 1 indicating that a destruction of the cakes takes place.

Two possible mechanisms for the cake destruction are discussed below.

Cracking during drying

It has been found, that drying can cause the build-up of inhomogeneous stresses due to the capillary forces, which in turn can crack a porous material [86]. Figure 4.15 shows the ratio of the Van der Waals forces to the capillary forces acting between two spherical alumina particles of diameter $d_p = 8$ nm at increasing S. The ratio has been calculated from the equations of appendix E.1 assuming the properties of cyclohexane for the liquid.

Figure 4.15 shows, that the capillary forces and Van der Waals forces generally are of the same order of magnitude. As expected the capillary forces become larger at lower S, however the Van der Waals forces are larger for all values of S. Since van der Waals forces are always acting between

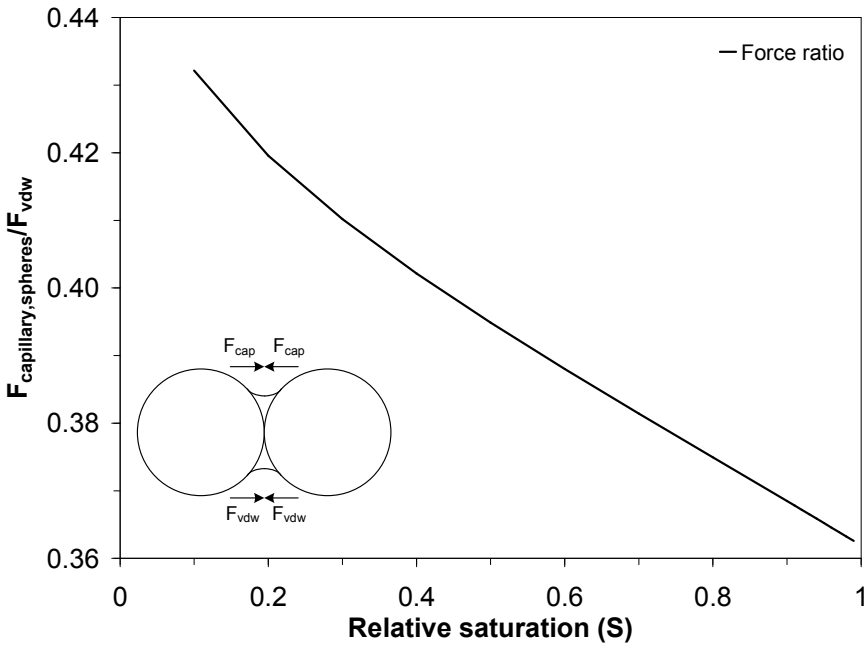


Figure 4.15 Ratio of the capillary force between two spherical alumina nanoparticles with diameter d_p = 8nm and the Van der Waals force for the same two particles as a function of the relative saturation S. The properties of the condensable gas is for cyclohexane.

the particles simultaneously with the capillary forces, it seems unlikely that destruction occurs due to an uneven distribution of capillary forces during drying.

Rearrangement of particles at high relative saturation

The Kelvin radius (r_k) is the radius of curvature for a liquid meniscus formed by condensation of a condensable species. It can be calculated from the Kelvin equation:

$$r_k = -\frac{2\sigma V_m}{RT \ln S} \tag{4.27}$$

where σ is the surface tension, and V_m the molar volume of the condensable species. It is possible for the liquid at high S to form a meniscus which is several times larger than the particle diameter. Therefore, the equations used for the calculations of the capillary force between two particles no longer apply. In order to calculate the capillary force caused by this meniscus, an assumption of the geometry is required. Assuming cylindrical capillaries and that the meniscus between these capillaries is hemispherical (an assumption often made in the determination of capillary-size distribution from capillary

condensation techniques [87, 88]), the Kelvin radius becomes equal to the cylindrical capillary size (diameter, d_{cc}), and the capillary force becomes:

$$F_{\text{capillary}} = 2\pi\sigma d_{cc} \tag{4.28}$$

Figure 4.16 shows the ratio of the capillary force to the Van der Waals force for two spherical alumina particles of diameter $d_p = 8$ nm, as well as the size of the cylindrical capillary size at which a liquid meniscus is formed, as a function of the relative saturation S. If one of the two particles is fixed, i.e. only one can move, the Van der Waals force is the smallest adhesion force the movable particle can experience. Particles situated in the cake typically have several neighbours (coordination number), which means that the minimum force required before separation occurs is most likely somewhat higher. Nonetheless, as the cake is highly porous, the coordination number is expected to be low.

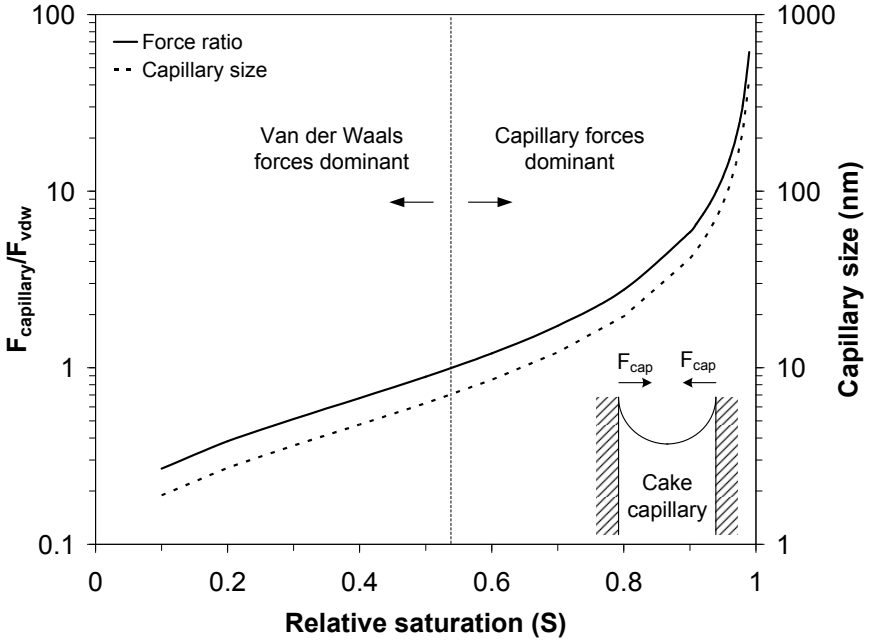


Figure 4.16 Ratio of the capillary forces to Van der Waals forces between two spherical alumina nanoparticles (diameter $d_p = 8$ nm) as a function of the relative saturation S. Also shown is the cylindrical capillary size at which a hemispherical meniscus forms.

At $S > 0.54$ the capillary force in the cake capillaries becomes larger than the Van der Waals force between two 8 nm particles. However, the size of the cake capillaries is in the same order of magnitude as the particle diameter, which may bring about changes on the nano-scale, however not on the micro-scale and it is unlikely that this will be detected by the permeability method.

The capillary force increases with increasing S and near $S = 1$ the force becomes much larger than the Van der Waals force between two particles. Since the forces inside the cake are not evenly distributed due to a size-distribution of the cake capillaries, it is possible for the surface tension to pull the particles together in order to minimize the surface energy in the largest capillaries. Furthermore, the sizes of the cake capillaries at which condensation occurs in increases to the order of 100 nm, which is the same order of magnitude as measured by permeability measurement and explains why the structural collapse is detected only at high S.

4.7.2 Sintered cakes

Figure 4.17 shows the permeability of a cake sintered at 700°C and exposed to various degrees of relative saturation of cyclohexane. It is clear from the figure, that the sintering has increased the mechanical strength of the cake, as no change in the permeability can be observed. The cake average capillary size determined by the permeability curves all varied within the experimental error of the method, indicating that no destruction had indeed taken place. An increase in the mechanical strength can only come through bonds between the particles, which are stronger than the Van der Waals bonds.

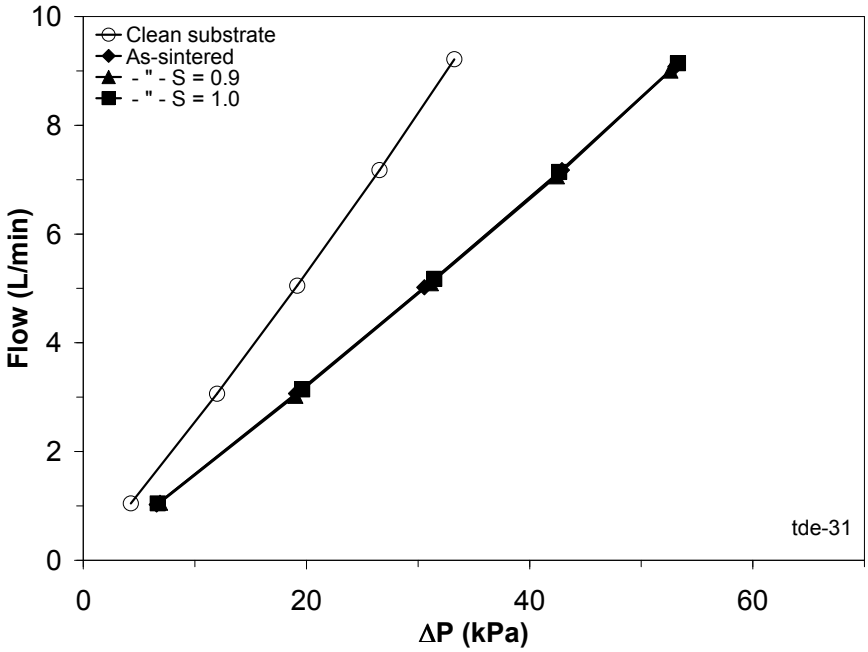


Figure 4.17 Permeability curves for an as-sintered cake exposed to several values of relative saturation of cyclohexane (S). Experiment label: tde-31. No change in the permeability is seen, indicating that the cake has increased in mechanical strength.

This indicates that sintering at 700°C has indeed produced sintering necks. TEM images should be captured on scraped-off cake particles to determine this definately. Furthermore, in line with the sintering study, experiments should also be carried out at 500°C and 900°C in future work.

4.8 Mechanical stability - conclusions

The mechanical stability of as-deposited and as-sintered cakes of alumina was investigated by condensing a gas (cyclohexane) near its saturation point ($S = 1$). The as-deposited cakes were seen to deteriorate and become more permeable between $S = 0.9$ and $S = 1.0$. This was most likely due to the condensation of material inside the porous cake structure, which yielded capillary forces orders of magnitude higher than the Van der Waals forces between two particles. As the capillaries in the cakes were not uniform, the forces would also be distributed unevenly eventually pulling apart particles in the largest capillaries and destroying the cake.

By sintering at 700°C it was shown, that the mechanical stability was increased markedly due to the formation of sinter necks, as the cake was not destroyed even at $S = 1$. No experiments were done at 500°C and 900°C, but a higher mechanical stability is to be expected as well.

In conclusion, to obtain mechanically stable cakes, it is a requirement that the constituent particles form sintering necks. Due to their small size, nanoparticles form sintering necks already at temperatures well below the densification temperature. This is an advantage, as the sintering of the highly porous cakes at higher temperatures (1100°C) was seen to destroy the layer due to significant shrinkage.

Chapter 5

Modelling deposition of nanoparticles

5.1 Introduction

Understanding the influence of deposition conditions on the morphology of particle films deposited on flat non-porous surfaces, has been the topic of several investigations [51, 53, 54, 89] (cf. section 3.2.2). In contrast, only few studies exists on the deposition on porous materials. Reis [90] modeled the deposition of coke particles inside a porous catalysts, assuming that deposition only occurred inside the porous structure. This forces the formation of artificially thin clogs at the capillary mouth, which never occur in real systems, as experimental studies have shown that deposition of nanoparticles form a homogeneous cake on top of a porous material, rather than penetrating the capillaries of that material [5, 15, 22].

An important process parameter in the constant flow filtration process on fibrous filters is the pressure-drop evolution [45, 46, 91, 92, 93]. Three regimes have been identified experimentally [93]: In the first regime, particles deposit within the filter and the increase in pressure-drop with mass-loading is quite slow. In the second regime, the captured particles start collecting incoming particles and plugging commences resulting in a faster build-up of the pressure-drop. In the third regime, the filter cake is formed and the pressure-drop increase rapidly.

Several numerical models have been set up to describe these regimes [91, 92, 94, 95]. In all cases, an assumption of the geometry of the deposit was required. Commonly, an effective fiber-diameter is used, which changes with time as particles deposits with constant solid volume fraction. Only few models actually allow for to describe all 3 regimes [92, 94]. Nevertheless, an assumption of cake formation time is required.

Here, deposition inside and outside of a capillary is studied by Langevin dynamics (LD) [51, 53, 64, 96, 97, 98]. From the detailed evolution of the

deposit morphology, the pressure-drop build-up is obtained. The model is validated in the limits of purely diffusional deposition (Pe $\rightarrow$ 0), purely ballistic deposition (Pe$\rightarrow \infty$), and most importantly with filtration theory. Structures grown for intermediate Pe numbers are also compared. For simplicity reasons, direct comparison of the model results with the ones obtained by filtration with fibrous filters rather than the porous ceramic substrates used in this work (see chapter 3), is done. This is due to the high porosity of the fibrous filter, which excludes heap filtration and simplifies the transition between the different regimes.

Finally, the link between the experimental part of this work is made. First, qualitatively by comparing the deposit morphology to the one obtained from SEM. Then, a quantitative comparison is made by comparing the plugging time and porosity to the experimentally obtained one.

5.2 Theory

5.2.1 Model overview

The porous substrate is modelled as cylindrical capillaries (fig. 5.1) in parallel to each other. Monodisperse particles are deposited by filtration through the capillaries. Particles are assumed to deposit one at a time [99]. In order to obtain true random motion before deposition, the particle is released at a certain distance from the capillary-mouth (fig. 5.1, drop domain). The particle flows into the capillary, due to Brownian motion and convective transport. The movement of the particle is determined by the solution of the Langevin equation of motion (cf. section 5.2.2). Particles only deposit by touching other particles or by touching the capillary-wall. If a particle escapes the domain on its way to the capillary it is reintroduced with an opposite velocity vector: open boundary [98]. Particles that penetrate through the capillary (e.g. reach the outlet without depositing) are counted. The deposits are allowed to grow in the whole deposition domain, in order to capture the change from capillary deposition to cake growth. The flow field is assumed to be plug-flow and the fluid-velocity is assumed to be constant in time, as the solid volume fraction of the deposits are expected to be low [53, 51].

5.2.2 Langevin equation of motion

The ordinary Langevin equation of motion is [96],

$$m\dot{\mathbf{v}} = -f\left(\mathbf{v} - \mathbf{w}\right) + \mathbf{F} + \mathbf{X} \tag{5.1}$$

where m is the particle mass, $\mathbf{v}$ is the particle velocity (vector), f is the friction factor (slip-correction included), $\mathbf{w}$ is the fluid velocity, $\mathbf{F}$ is the external force, and $\mathbf{X}$ is a random force (due to Brownian motion). Assuming

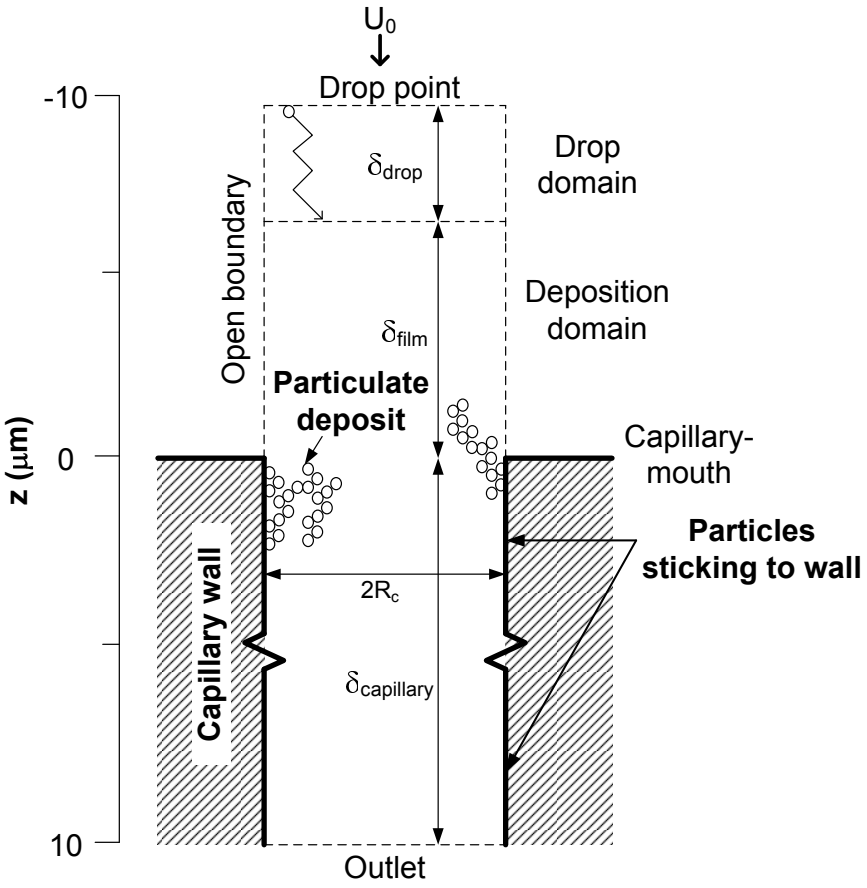


Figure 5.1 Schematic of the simulated domain and boundary conditions. Particles are inserted at the top (drop point) and flow downstream (velocity U_0). First through the drop domain (δ_{drop}) where deposition is not allowed in order to assure true random motion. They continue in the deposition domain where they deposit by colliding either with previously deposited particles or the capillary walls. If a particle reaches the outlet without collision it exits the domain, and a new particle is inserted at the top.

no external forces are acting on the particles, partial integration of eq. 5.1 yields [97]:

$$\begin{aligned}\mathbf{v}(t+\Delta t) &= \mathbf{V} + \mathbf{v}(t)\exp^{-\beta\Delta t} + \mathbf{w}\left(1-\exp^{-\beta\Delta t}\right)\\ \mathbf{r}(t+\Delta t) &= \mathbf{R} + \frac{\mathbf{v}(t)}{\beta}\left(1-\exp^{-\beta\Delta t}\right) + \mathbf{w}\left(\Delta t - \frac{1}{\beta}\left(1-\exp^{-\beta\Delta t}\right)\right)\end{aligned} \tag{5.2}$$

where $\mathbf{r}$ is the particle position vector, $\beta = f/m$, while $\mathbf{V}$ and $\mathbf{R}$ are the velocity and particle displacement vectors respectively, due to the random force. The time-step used in the integration of the Langevin equation, is chosen such that the value of the root mean square displacement due to diffusion of the particles is equal to 50% of the particle diameter [98]. If the time-step is large and the fluid velocity is high, the time-step is calculated from the average fluid velocity instead, allowing a maximum displacement of 50% of the particle diameter in the axial direction to take place. This choice of time-step is in comparison to that used by Ermak and Buckholz [96] and has been validated by calculating the diffusion coefficient, which has less than 5% difference from the one obtained directly from the Einstein equation (including slip-correction).

5.2.3 Equations for particle-particle collision

Figure 5.2 shows a particle in motion from position 0 to position e. By particle position, one refers always to its *center*. The moving particle has initial coordinates $\mathbf{x}_0 = (x_0, y_0, z_0)$. Its end coordinates are $\mathbf{x}_e = (x_e, y_e, z_e)$. The particle is assumed to move in a straight line from initial position to end position. In the example on figure 5.2, the particle will collide with a deposited particle with coordinates $\mathbf{x}_d = (x_d, y_d, z_d)$.

Parametric equations are used in order to check for the particle-particle collisions and to calculate the resulting collision coordinates, between a moving particle and a deposited one. In order to minimize the number of calculations required, a neighbour list is build from a grid-system. Collisions are only checked with the particles in the neighbour list. The parametric equation for the line $\mathbf{x}_0 \bar{-} \mathbf{x}_e$ is:

$$\mathbf{p} = (\mathbf{x}_e - \mathbf{x}_0)t + \mathbf{x}_0 \tag{5.3}$$

where t is a parameter indicating the position on the line. It is a requirement for collision, that the minimum distance $|d_{\min}|$ between any particle (its center) and the line $\mathbf{x}_0 \bar{-} \mathbf{x}_e$, is less than the sum of their radii, or, since monodisperse particles are used, 1 particle diameter:

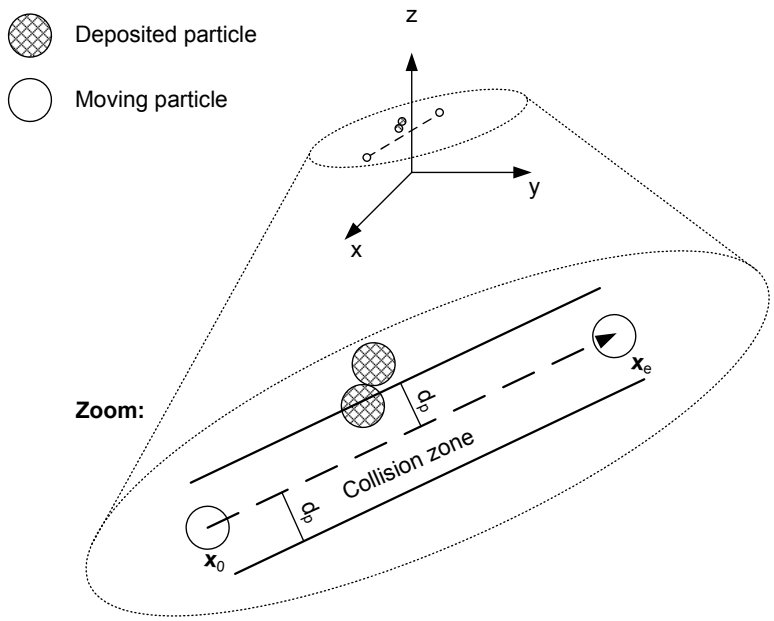


Figure 5.2 Particle motion from $\mathbf{x}_0$ to $\mathbf{x}_e$ with resulting collision. The distance traveled has been greatly exaggerated. Deposited particles are marked as filled.

$$|d_{\min}| = \frac{(\mathbf{x}_e - \mathbf{x}_0) \times (\mathbf{x}_0 - \mathbf{x}_d)}{|\mathbf{x}_e - \mathbf{x}_0|} \leq d_p \tag{5.4}$$

where $\times$ denotes the vectorial cross-product. Furthermore, it is required that the distance the particle travels on the parametric line is not greater than $t = 1$, as it would otherwise have moved longer than e:

$$t_c \leq 1 \tag{5.5}$$

where t_c is the value of the parameter t at the point of collision. Collision occurs if both 5.4 and 5.5 are fulfilled for a given $\mathbf{x}_d$. The collision parameter t_c is found, since in the case of collision, the distance between the collision point (on the $\mathbf{x}_0 \bar{-} \mathbf{x}_e$ line) and the colliding particle is equal to the sum of the particle radii (deposited and moving particle). For monodisperse particles where both particle radii are identical, the distance is d_p:

$$d_p = |\mathbf{x}_c - \mathbf{x}_d| = |(\mathbf{x}_e - \mathbf{x}_0)t_c + \mathbf{x}_0 - \mathbf{x}_d| \tag{5.6}$$

or:

$$d_p = \sqrt{((x_e - x_0)\, t_c + (x_0 - x_d))^2 + ((y_e - y_0)\, t_c + (y_0 - y_d))^2 + ((z_e - z_0)\, t_c + (z_0 - z_d))^2} \tag{5.7}$$

which can be solved for t_c:

$$t_c = min\left(\frac{-1 \pm \sqrt{b^2 - 4ac}}{2a}\right) \quad , \quad t_c > 0 \tag{5.8}$$

where min denotes the t_c which is smallest of the two solutions, however always > 0. The constants a, b and c are given by:

$$\begin{aligned} a &= |\mathbf{x_e} - \mathbf{x_0}|^2 \\ b &= 2\,(\mathbf{x_e} - \mathbf{x_0}) \cdot (\mathbf{x_0} - \mathbf{x_d}) \\ c &= |\mathbf{x_0} - \mathbf{x_d}|^2 - d_p^2 \end{aligned} \tag{5.9}$$

Once t_c is known, the collision coordinates are simply calculated from eq. 5.3:

$$\begin{aligned} x_c &= (x_e - x_0)t_c + x_0 \\ y_c &= (y_e - y_0)t_c + y_0 \\ z_c &= (z_e - z_0)t_c + z_0 \end{aligned} \tag{5.10}$$

5.2.4 Equations for particle-wall collision

If the particle can collide with the capillary wall, it is required that:

$$x_e^2 + y_e^2 \geq R_c - 0.5d_p \tag{5.11}$$

where R_c is the radius of the capillary. If eq. 5.11 is fulfilled, then collision occurs exactly at one particle radius ($0.5d_p$) away from the capillary wall due to interception:

$$x_c^2 - y_c^2 = R_c - 0.5d_p \tag{5.12}$$

Inserting the parametric equation of the line (eq. 5.3) in eq. 5.12 gives:

$$((x_e - x_0)t_{wc} + x_0)^2 + ((y_e - y_0)t_{wc} + y_0)^2 = (R_p - 0.5d_p)^2 \tag{5.13}$$

where t_{wc} denotes the value of parameter t at the point of collision with the wall. Equation 5.13 can be solved for t_{wc}:

$$t_{wc} = min\left(\frac{-b_w \pm \sqrt{b_w^2 - 4a_w c_w}}{2a_w}\right) \quad , \quad t_{wc} > 0 \tag{5.14}$$

where min denotes the t_{wc} which is smallest of the two solutions. As for t_c it is required that it is always > 0. The constants a_w, b_w and c_w are given by:

$$\begin{aligned} a_w &= (x_e - x_0)^2 + (y_e - y_0)^2 \\ b_w &= 2\left((x_e - x_0) \cdot x_0 + (y_e - y_0) \cdot y_0\right) \\ c_w &= x_0^2 + y_0^2 - (R_c - 0.5d_p)^2 \end{aligned} \tag{5.15}$$

Once t_{wc} is known, the collision coordinates are found from eq. 5.3:

$$\begin{aligned} x_c &= (x_e - x_0)t_{wc} + x_0 \\ y_c &= (y_e - y_0)t_{wc} + y_0 \\ z_c &= (z_e - z_0)t_{wc} + z_0 \end{aligned} \tag{5.16}$$

5.2.5 Multiple collisions

In some cases multiple collisions are possible. Furthermore, the particle may collide with the wall rather than another particle. If the number of possible collisions is n (including both particle-particle collision and particle-wall collision), then the corresponding set of possible collision parameters is $t_{c,1}, t_{c,2}, ..., t_{c,n}$. Since only one collision can occur due to sticking, the one which is hit is simply the first one, or in terms of t_c:

$$t_c = min(t_{c,1}, t_{c,2}, ..., t_{c,n}) \tag{5.17}$$

5.2.6 Deposition program

The code was programmed in Fortran 90, and the full code including the comments, is given in appendix F.3. The program structure is further shown in appendix G.1, where a detailed calculation flow-sheet is presented.

5.2.7 Deposition on flat and porous substrates

For validation purposes, the deposition of particles on flat (non-porous) substrates was studied. The model domain was modified by removing the capillary walls and applying an open boundary condition over the entire domain. Sticking was allowed only to take place at the flat surface and to already deposited particles. The standard conditions can be seen in table 5.1. The size of domain (given in terms of particle size, d_p) is similar to that of previous works [53, 10]. The deposits have been characterized by their fractal dimension (D_f) and compared to literature data [53, 100].

The deposition of particles in the capillaries of the porous substrate, was carried out as shown in figure 5.1. The conditions for deposition can be seen in table 5.1. A particle number-concentration of 10^{14} #/m^3 was used throughout the calculations. The value is in the same order of magnitude as the concentration of the aerosol used in this work (see table 5.2). The deposit was grown to a maximum height of 9 μm in order to investigate the evolution of the morphology with time. The deposits were characterized qualitatively as well as quantitatively by calculating the solid volume fraction and the plugging time.

Table 5.1 Model parameters for particle deposition.

Substrate type	Particle diameter (d_p)	Radius (R_c)	δ_{film}	$\delta_{\text{capillary}}$	δ_{drop}
Flat	25, 50, 100 nm	$40d_p$	$80d_p$	-	$20d_p$
Porous	50 nm	1 - 4 μm	9 μm	10 μm	1 μm

Solid volume fraction

A common way of characterizing the morphology of a particle deposit is through its solid volume fraction (ϕ_s) [54]. This is found by calculating the ratio of particle volume within a slice of thickness d_p (1 particle diameter) to the total volume of the slice ($\pi R_c^2 d_p$):

$$\phi_s = \frac{V_p(z)}{\pi R_c^2 d_p} \tag{5.18}$$

where $V_p(z)$ is the total volume of particles contained in the slice at height z and thickness d_p.

Plugging time

Initially, particle deposition takes place inside the capillary then proceeds to form a cake outside of the capillary. The transition from one regime to

the other is characterized by the plugging time. The fraction of particles entering through the capillary mouth is:

$$P_E = \frac{\dot{N}_{cap}}{\dot{N}_{ins}} \tag{5.19}$$

where $\dot{N}_{cap}$ is the flow of particles (#/s) entering the capillary and $\dot{N}_{ins}$ is the flow of particles entering the deposition domain (fig. 5.1). In this work, the plugging time t_{plug} is defined as the time required until P_E is lower than 0.1 %.

5.2.8 Pressure-drop evolution

A commonly used filter material is the fibrous filter. This filter type consists of several woven fibers, interconnected to form a highly porous filter with a large filtration surface area to capture the particles. Here, the fiber filter structure is assumed equivalent to that of several equally spaced capillaries, where the wall of the capillary correspond to the fibers surrounding it. The distance between the capillaries is assumed negligible due to the high porosity of the filter. Therefore, the gas velocity U_0 in the capillary becomes equal to the gas velocity at the filter surface (face velocity). Inside the capillary, the pressure-drop is calculated from the Hagen-Poiseuelle equation for a compressible gas:

$$Q(z) = -\frac{\pi R_{c,eff}^4(t,z)}{8\mu_g}\frac{\mathrm{d}P}{\mathrm{d}z} \tag{5.20}$$

where dP/dz is the pressure gradient (Pa/m), μ_g is the gas viscosity (kg/ms) and $R_{c,eff}(t,z)$ (m) is the effective capillary radius. This value is related to the initial capillary radius through a mass-balance with the deposited particles:

$$R_c^2(1-\phi_s(t,z)) = R_{c,eff}^2(t,z) \tag{5.21}$$

Eq. 5.21 has been derived assuming a uniform particle coating in each slice dz as in model studies for Nuclepore filters [101, 48]. Rewriting eq. 5.20 in terms of the molar flow ($\dot{n}$, moles/s) is more favorable, since this value is independent on the depth as opposed to Q, which changes due to the compression:

$$\frac{\dot{n}RT}{P} = -\frac{\pi(1-\phi_s(t,z))^2 R_c^4}{8\mu_g}\frac{\mathrm{d}P}{\mathrm{d}z} \tag{5.22}$$

where R is the gas constant (J/mol K) and T is the temperature (K). The flow-rate at pressure $P_0 = 1$ bar is $Q_0 = \frac{\dot{n}RT}{P_0} = U_0 \pi R_c^2$, which is the constant flow at the filter-face. One can then write eq. 5.22 as:

$$P\frac{\mathrm{d}P}{\mathrm{d}z} = -\frac{8\mu_g U_0 P_0}{(1-\phi_s(t,z))^2 R_c} \tag{5.23}$$

As the solid volume fraction profile, $\phi_s(t,z)$, is known from the LD computations, eq. 5.23 can be solved numerically. The pressure-drop through the deposit which grows on the outside of the capillary, is modeled using a slightly modified version of d'Arcy's law, as the structure of this deposit resembles more that of film rather than of a capillary. The molar flow through a slice dz is given by [56]:

$$\dot{n} = -\frac{\pi R_c^2 B_0(t,z)}{\mu_g RT} P\frac{\mathrm{d}P}{\mathrm{d}z} \tag{5.24}$$

where $B_0(t,z)$ (m^2) is the permeability at time t and depth z. Introducing Q_0, eq. 5.24 becomes:

$$P\frac{\mathrm{d}P}{\mathrm{d}z} = -\frac{\mu_g Q_0 P_0}{\pi R_c^2 B_0(t,z)} \tag{5.25}$$

The permeability $B_0(t,z)$ depends on the capillary-size of the deposit (d_{cc}) and the tortuosity factor (τ) according to [58] by:

$$B_0(t,z) = \frac{d_{cc}^2(t,z)}{32}\left(\frac{1-\phi_s}{\tau}\right)(t,z) \tag{5.26}$$

For low values of ϕ_s the tortuosity factor approaches unity (see eq. 3.26). The capillary-size of the deposit is commonly expressed in terms of particle size and solid volume fraction as [62]:

$$\frac{d_{cc}}{d_p} = \frac{2}{3}\frac{1-\phi_s(t,z)}{\phi_s(t,z)} \tag{5.27}$$

Inserting 5.27 in 5.26 gives an expression for the permeability in terms of particle size d_p and solid volume fraction ϕ_s:

$$B_0(t,z) = \frac{d_p^2}{72}\frac{(1-\phi_s(t,z))^3}{\phi_s^2(t,z)} \tag{5.28}$$

Finally, the expression for the pressure-drop through the deposit is found by insertion of eq. 5.28 into 5.25:

$$P\frac{\mathrm{d}P}{\mathrm{d}z} = -\frac{72\phi_s^2(t,z)\mu_g Q_0 P_0}{\pi R_c^2 d_p^2(1-\phi_s(t,z))^3} \tag{5.29}$$

From eq. 5.23 and 5.29 the total pressure drop can be found numerically.

Classical filtration theory

After capillary plugging, a cake with a constant solid volume fraction builds up as in classic filtration theory. For constant ϕ_s, the pressure-drop evolution with time can be found analytically [102]:

$$\Delta P_{cake}(t) = P_0 - \sqrt{P_0^2 - \frac{144\mu_g U_0^2 P_0 \phi_{s,c}}{d_p^2(1-\phi_{s,c})^3} C_n v_p (t - t_{plug})} \quad , \quad t > t_{plug} \tag{5.30}$$

where $\phi_{s,c}$ is the constant solid volume fraction in the cake, t_{plug} is the plugging time, v_p is the particle volume, and t is the actual filtration time. The total pressure-drop is found by adding the pressure-drop through the initial deposit, which has been deposited until the onset of plugging, $\Delta P_{plug}(t)$, to that of the cake pressure-drop, $\Delta P_{cake}(t)$. Assuming that the structure of the initial deposit does not change after plugging, the final expression for ΔP can be shown to be:

$$\Delta P(t) = P_0 - \sqrt{(P_0 - \Delta P_{cake}(t))^2 - 2\kappa} \tag{5.31}$$

where the constant κ (Pa^2) depends on the structure of the initial deposit:

$$\kappa = \frac{1}{2}\left(P_0^2 - (P_0 - \Delta P_{plug}(t_{plug}))^2\right) \tag{5.32}$$

where $\Delta P_{plug}(t_{plug})$ is the pressure-drop at t_{plug}. The derivation of 5.30, 5.31, and 5.32 can be seen in appendix F.1, where also equations for calculating the flow evolution for constant pressure-drop filtration has been shown. However, due to the time limitation of the project, it was not possible to implement these.

5.3 Results and discussion

5.3.1 Model validation

Figure 5.3 shows a grown deposit for 25 nm particles at Pe = 0.001. The structure of the deposit is seen to be fractal-like with a low solid volume

fraction ($\phi_s = 0.02$). The fractal dimension of the deposits for several Pe numbers is reported on figure 5.4. The results (circles) have been compared to the ones obtained by Kulkarni and Biswas (triangles) [53] and for pure diffusionally grown deposits (line) [103]. Good correlation between literature and present model results can be observed. The fractal dimension obtained for the different particle sizes (25 - 100 nm) varied less than 5% at any given Pe number. To save computational time, a particle-size of 50 nm was chosen for the porous surface calculations further on.

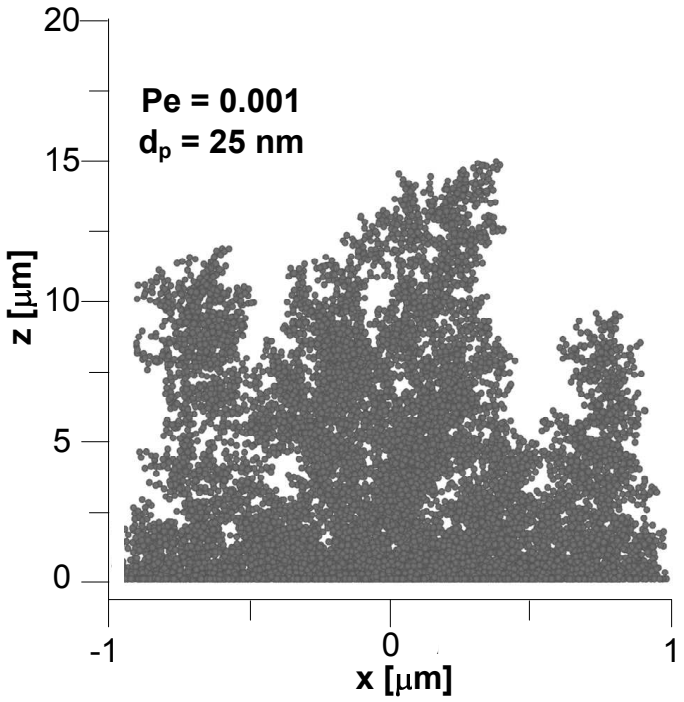


Figure 5.3 Side-view of a film deposited on a flat surface. The film has low solid volume fraction ($\phi_s \approxeq 0.02$) and show a fractal like structure.

When the particles stick to the walls but do not change the capillary size of the substrate, this is known as "perfect sink deposition" [29]. An analytical solution to this problem can be found by solving the diffusion-convection equation for large Pe numbers (Pe $>$ 100). On figure 5.5 the results of the perfect sink model (for plug-flow) have been compared to the Langevin dynamics model with perfect-sink boundary conditions.

Two runs (cf. fig. 5.5) show good comparison between the numerical and analytical results, with less than 5% difference for all values of μ (dimensionless length).

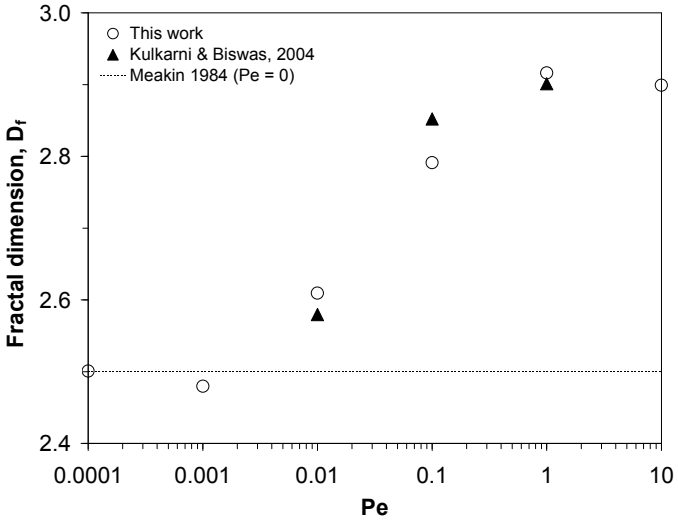


Figure 5.4 Fractal dimension of films deposited on flat surfaces as a function of the Pe number. The fractal dimension (D_f) follows the typical S-shape with flattening at low and high Pe numbers.

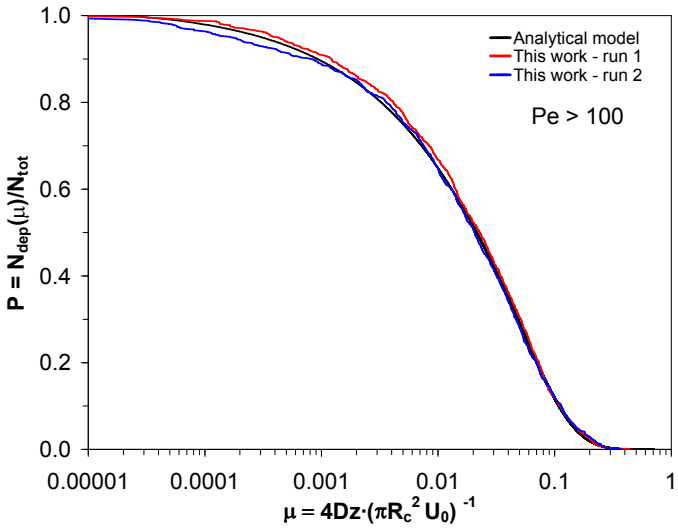


Figure 5.5 Analytic solution of the perfect-sink model for plug-flow and comparison to the results of the LD simulations with perfect-sink boundary conditions. The LD solution and the analytical model agree within <5% for all μ values.

5.3.2 Porous surface deposition

Deposition dynamics

Figure 5.6a,b shows the development of a deposit structure at Pe = 1 (d_p = 50 nm, U_0 = 0.1 m/s) and R_c = 2 μm. The capillary gradually closes (a) as the effective radius is decreased from R_c to 0. The deposit grows from the capillary mouth (b) forcing most of the particles to be deposited outside of the capillary. At t = 630 s (b), the capillary is considered to be plugged as practically no particles penetrate through the capillary mouth after this time ($P_E \approx 0$). Further deposition of particles (b, 1400 s) lead to cake growth and only extend the deposit upstream of the flow.

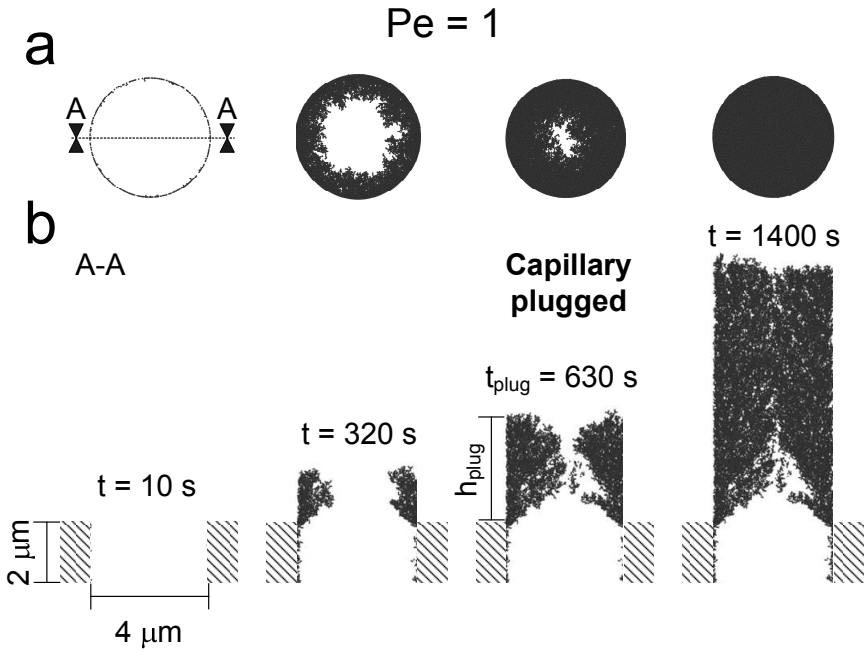


Figure 5.6 Topview (a) and cross-sectional cut (b) of a particle deposit at Pe = 1 and $R_c = 2\mu$m. The deposition time is indicated above the deposit. The deposit grows mainly from the capillary-mouth and outwards.

Figure 5.7(a) shows the evolution of the solid volume fraction through the model domain with time for Pe =1 with R_c = 2 μm. Deposition starts near the capillary-mouth in agreement with figure 5.6. At 10 s (fig. 5.7b) there is a small increase of ϕ_s near the capillary-mouth from particles accumulating there. The solid volume fraction profile gradually builds up, ending in a constant value ($\phi_{s,c}$), which is also typical in flat surface deposition [54]. After 320 s (figure 5.7, red line), the profile changes only slightly downstream of the solid volume fraction maximum. This was expected from figure 5.6 where further deposition was seen to modify only a part of the morphology below the deposit height. The reason is, that particles deposit mostly near

the first available deposition surface. However, it is not until after 630 s (b), that particles no longer penetrate the capillary and the solid volume fraction inside the capillary becomes constant. Once the capillary is plugged (630 s), a constant solid volume fraction is obtained throughout the cake (figure 5.7, 1000 s, 1400 s). This shows that classical filtration theory should be applicable after the time of capillary plugging.

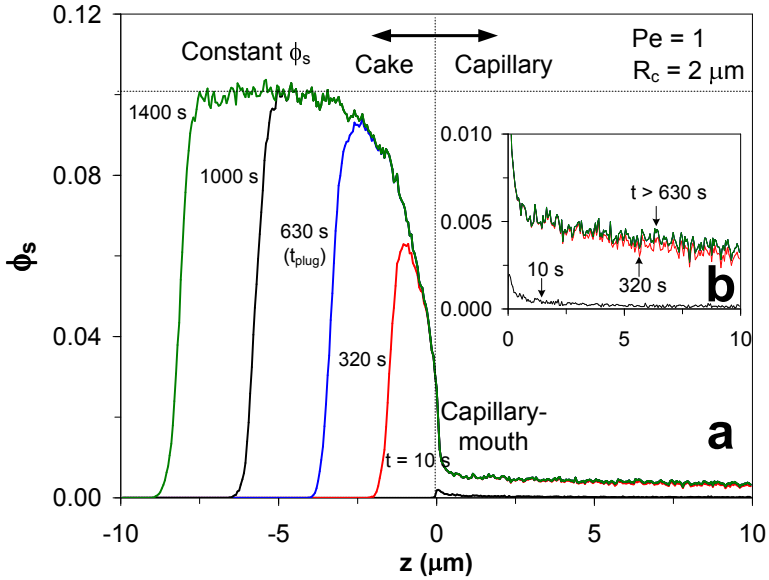


Figure 5.7 Evolution of solid volume fraction profile with time (a) (10 - 1400 s) for Pe = 1, $R_c = 2\mu$m. Deposition takes place initially mainly near the capillary-mouth (10 s, black line). Nearly all deposition occurs subsequently outside of the capillary (320 s, red line). After plugging (630 s, blue line), a constant solid volume fraction is reached (1400 s, green line). An enlargement (b) of the profile inside the capillary reveals that the solid volume fraction inside the capillary becomes constant after 630 s (t_{plug})

Figure 5.8 shows the entering particle fraction at the capillary-mouth as a function of time for several Pe numbers. Three regimes (a-c) are defined. First capillary deposition (a): all particles enter through the capillary mouth. Then capillary plugging (b): deposition near the capillary-mouth starts decreasing the particle free area, which accelerates the deposition of particles outside of the capillary. Finally cake growth (c): no particles penetrate the capillary and a cake of constant solid volume fraction builds up as expected from figure 5.7.

Figure 5.9 shows the particle penetration profiles for Pe = 0.1 (a) and Pe = 1 (b) respectively. For Pe = 0.1 (a), the penetration profile is accurately described up to 100 s using the perfect-sink model. This is expected as the plugging has not yet fully begun (cf. fig. 5.8). At the plugging time (fig. 5.9a, blue line) the penetration profile differs from that of the perfect-sink model. This is because particles that deposit near the capillary-mouth (fig.

5.8b) shade off the rest of the capillary. Only particles outside of the shaded area (center of capillary) may penetrate. These are the particles which have the longest distance to diffuse to the capillary walls. This increases the depth at which they will deposit, thereby shifting the penetration profile towards that of larger penetration. The shading effect increases with increasing Pe numbers (fig. 5.9b). The perfect-sink model is therefore not valid when significant shading takes place, in contrast to the assumptions used in many filtration models that describe the dynamics of filtration during plugging [39, 95, 92].

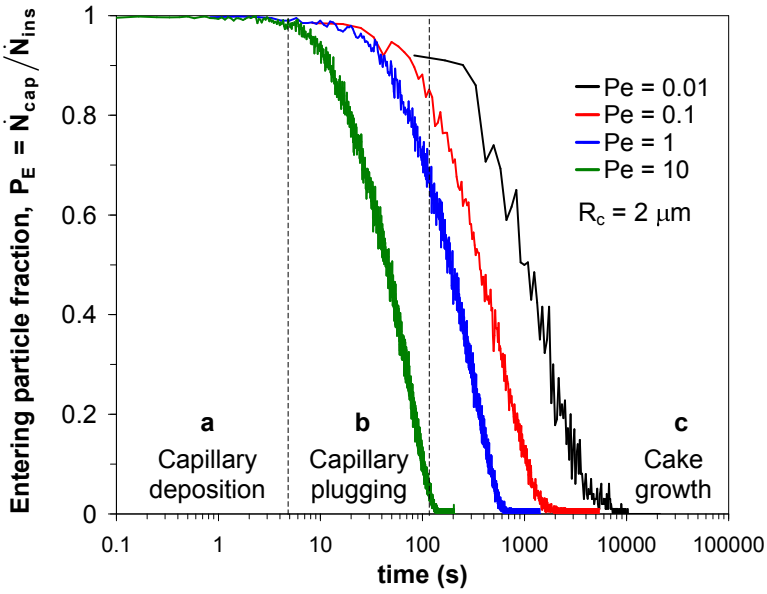


Figure 5.8 Fraction of total flux ($\dot{N}_{ins}$) entering through capillary-mouth ($\dot{N}_{cap}$) for several Pe numbers as a function of time for $R_c = 2\mu$m. Three regimes are observed (example here for Pe = 10): All particles penetrate through the capillary-mouth i.e. pure capillary deposition (a). Capillary becomes plugged (b) and more and more particles deposit outside of the capillary. Capillary is now totally plugged (c) and no more particles can penetrate, leading to only cake growth.

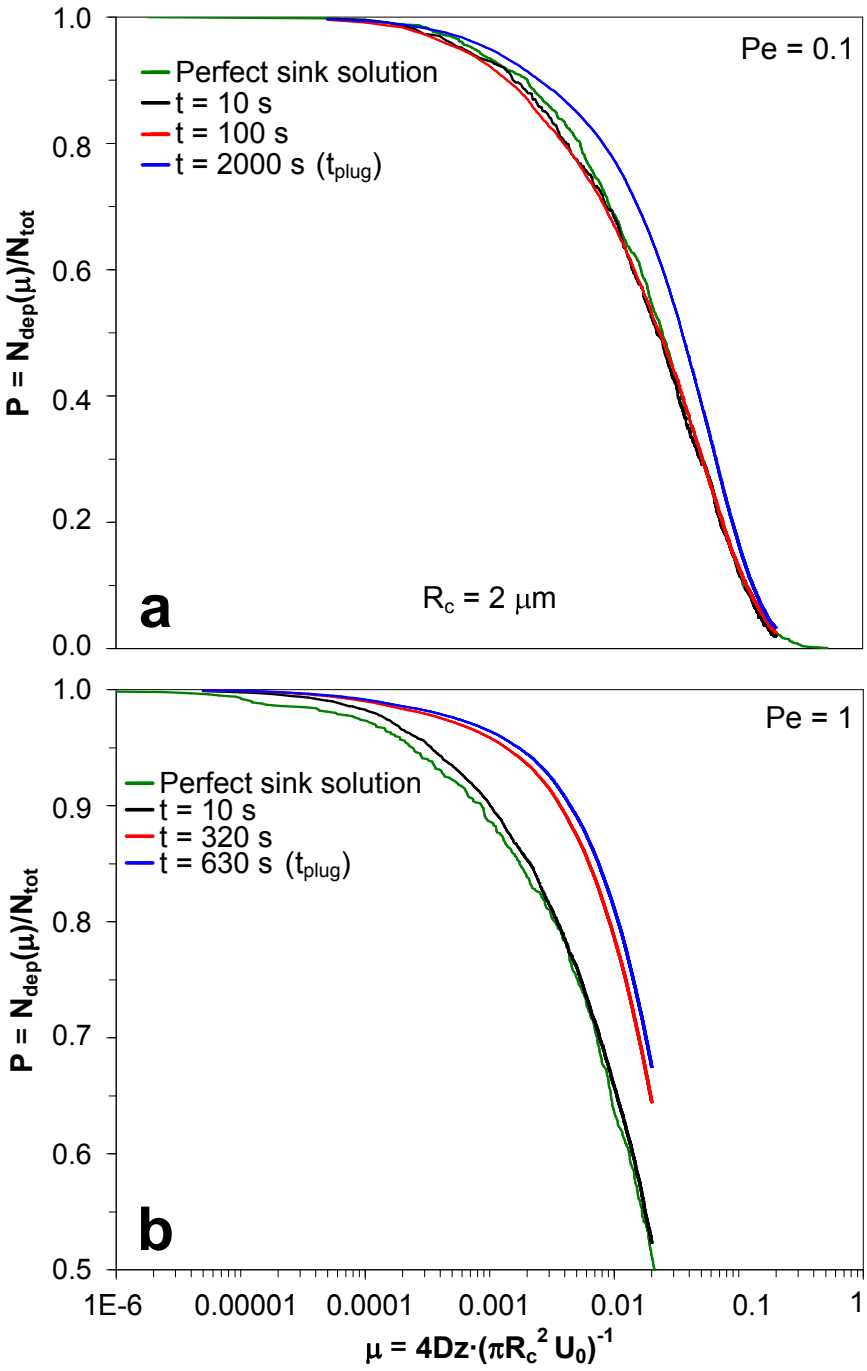


Figure 5.9 Penetration profile at Pe = 0.1 (a) for several times during capillary plugging. The perfect sink solution is shown to be sufficiently accurate up to 100 s, while once the capillary is fully plugged (2000 s), it fails to predict particle penetration. The increased penetration is a result of the shading effect of the capillary. The shading effect becomes even more pronounced at Pe = 1 (b).

Characterization of the morphology

Figure 5.10 shows the structure of the deposits at $R_c = 2\mu$m for several Pe numbers, at the time of capillary plugging. Increasing the Pe number increases proportionally the fluid velocity U_0, since the particle size is kept constant. Plugging takes place mainly outside of the capillary, as approximately 83 - 60% (Pe = 0.01 - 10) of the particle mass is deposited there.

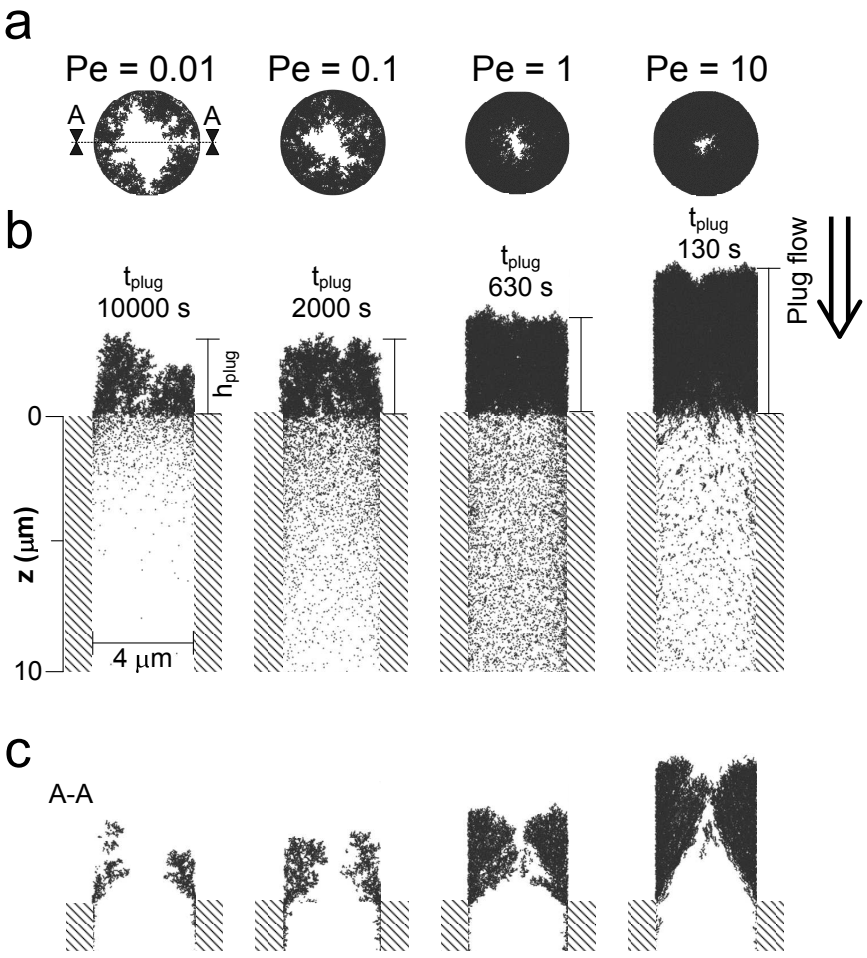


Figure 5.10 Top-view (a) of the deposits formed by filtration of 50 nm particles at the time of capillary plugging for several Pe numbers. At plugging, the particle free area decreases (a). For Pe = 0.01, the reduction is 75%, while at Pe = 10 the reduction is nearly 100%. The plugging height (b, h_{plug}) and solid volume fraction increase with increasing Pe, along with an increase in particle penetration through the outlet of the capillary. Cross-sectional cuts (c) reveals, that particle deposition takes place mainly outside of the capillary. The shape of the deposit at higher Pe numbers resembles that of a void cone.

The time required for plugging decreases from 10000 s to 130 s with Pe

increasing from 0.01 to 10. Increasing the Pe from 0.01 to 10, increases the particle flux by 3 orders of magnitude, however plugging time only decreases by 2 orders of magnitude. The reason is that the solid volume fraction increases with increasing Pe numbers allowing more particles to deposit before plugging.

The height of the deposit at capillary plugging (fig. 5.10b, h_{plug}) increases from 3.0 μm to 5.7 μm with increasing Pe from 0.01 to 10. At plugging, the particle free area (a) is decreased by 75% at Pe = 0.01, while at Pe = 10 this reduction is nearly 100%. This is an effect of larger residence time at low Pe number, which allows particles to be captured by diffusion within the deposit before penetrating through the capillary mouth, even at relatively high particle free area. With increasing Pe number, a void cone-like structure (fig. 5.10c) builds up. At Pe = 1 and Pe = 10, the tip of the cone coincides well with the plugging height (h_{plug}). Therefore, the plugging height can be estimated from the shape of the cone.

Figure 5.11a-b shows the solid volume fraction profiles for deposits formed at Pe = 0.01 (a) and Pe = 10 (b) respectively with a film height (δ_{film}) of 9 μm for several R_c (1 - 4 μm). At low Pe (Pe = 0.01, fig. 5.11a), deposition occurs with a maximum solid volume fraction at the capillary-mouth, whereas at high Pe (Pe = 10, fig. 5.11b), the maximum is obtained in the cake. Regardless of the radius and the Pe number, a constant solid volume fraction is obtained similar to flat surface deposition [54, 89]. The height above the capillary mouth at which the constant value is obtained, increases from 2.5 to 6 μm for $R_c = 2$ μm when Pe increases from 0.01 to 10. This is in good agreement with the plugging height, which increased from 3.0 to 5.7 μm for the same values of the Pe number. Therefore, as expected from figure 5.7, the constant ϕ_s ($\phi_{s,c}$) is reached only if the capillary is plugged. This means, that at Pe = 10 and $R_c = 4$ μm, the capillary is not yet plugged.

Figure 5.12 shows the constant solid volume fraction along with values obtained for flat surface deposition in the literature [54]. The ballistic limit value obtained experimentally as well as numerically by Blum and Schräpler [55] for ballistically limited deposition is shown as the reference asymptotic value (broken line).

Similar trends for these two different geometries are seen. In fact, at low Pe (Pe = 0.01) the capillary walls behave as a flat surface as the particles have no preferred velocity (diffusion limited). At high Pe however, the capillary walls play a minor role in determining the final morphology, due to the high ratio of the vertical to the horizontal flux of particles.

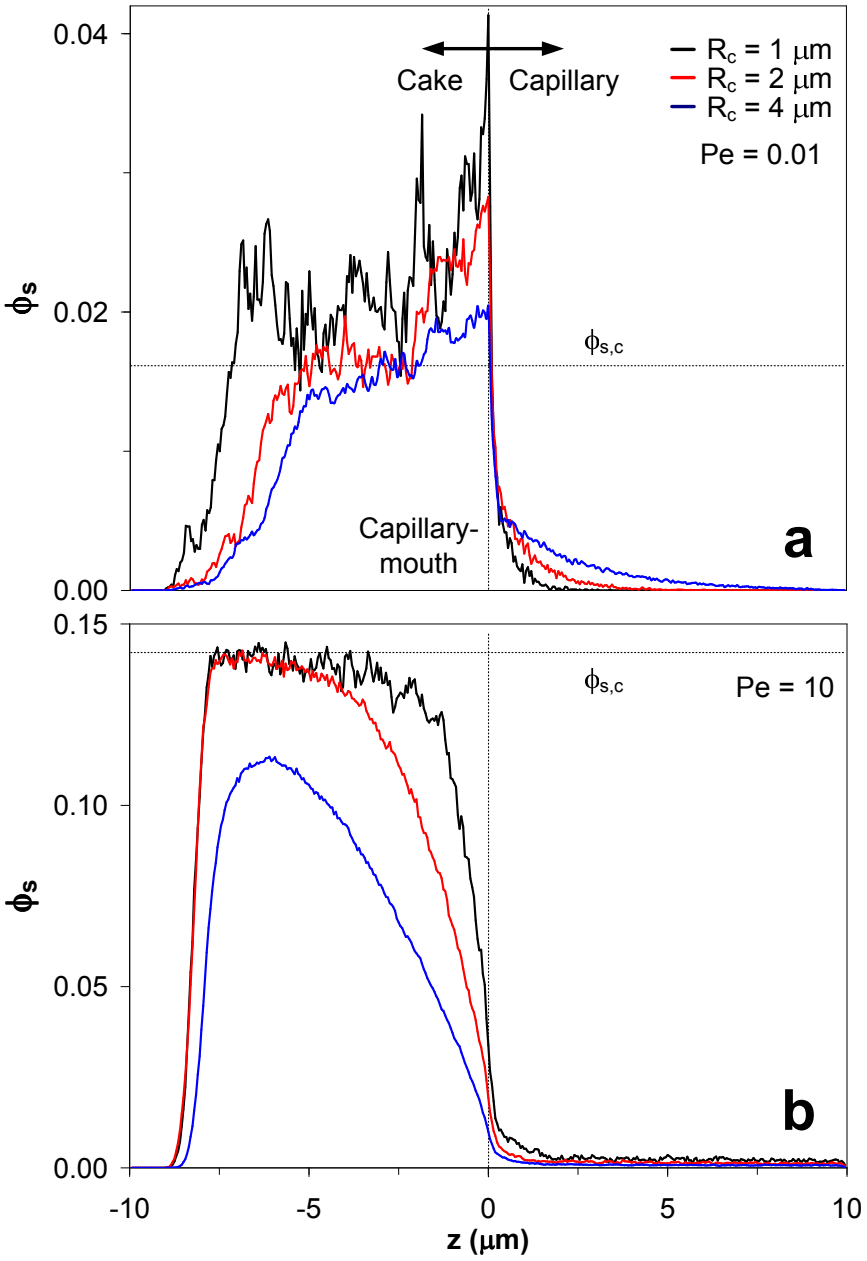


Figure 5.11 Solid volume fraction profile through cake and capillary. For Pe = 0.01 (a) the solid volume fraction increases sharply inside the capillary and reaches a maximum at the capillary-mouth. In the cake, it reaches a constant value. For Pe = 10 (b) the maximum (and constant) value is reached in the cake. If the capillary is not plugged (b, $R_c = 4$ μm), the constant solid volume fraction is not reached. The constant solid volume fraction is approximately the same regardless of the capillary radius, indicating that it is independent on the geometry.

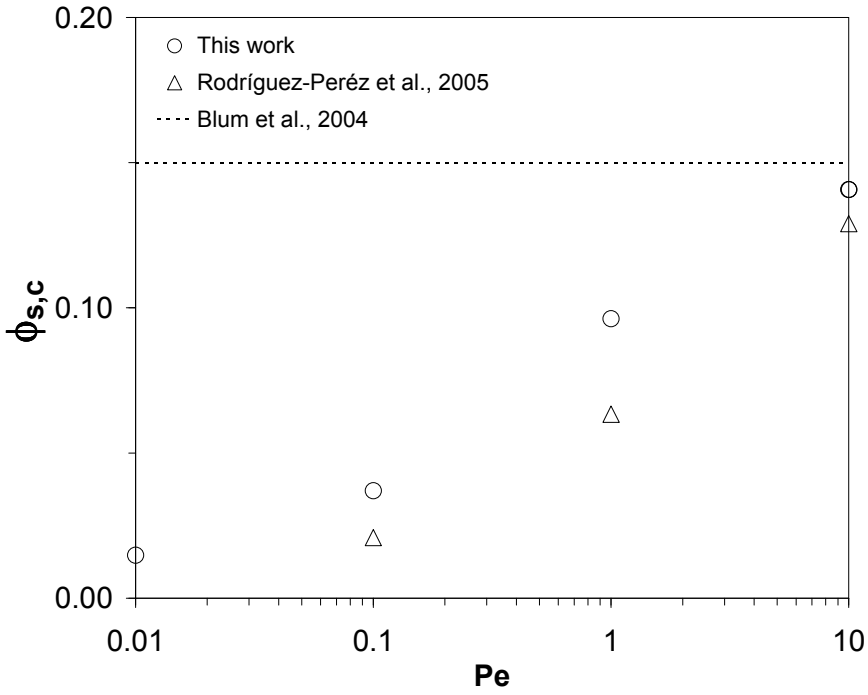


Figure 5.12 Constant solid volume fraction as a function of the Pe number. The values have been compared to the ones obtained by Rodríguez-Peréz et al. 2005 [54] for flat surface deposition. The constant solid volume fraction increases with increasing Pe, which is in good agreement with the compact structure obtained at high Pe.

5.3.3 Pressure-drop evolution

Figure 5.13 shows the pressure-drop evolution at Pe = 1 with $R_c = 2$ μm. The shape of the curve agrees qualitatively well with the ones obtained experimentally in the literature [45, 92]. During capillary deposition, the pressure-drop is low and the increase is slow, which shows that capillary deposition does not sufficiently contribute to the pressure-drop evolution. The pressure-drop starts increasing more rapidly when particles deposit both inside and outside of the capillary during capillary plugging.

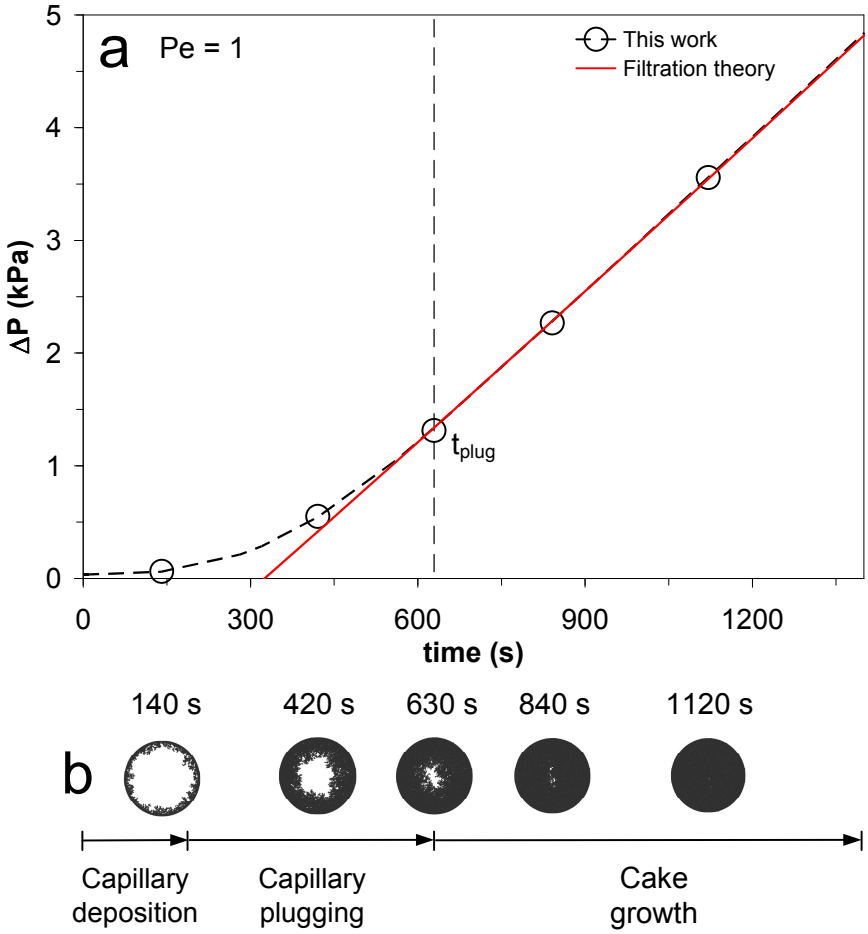


Figure 5.13 Pressure-drop (a, circles) at Pe = 1 as a function of time for a fibrous filter (average capillary-size, $R_c = 2\mu$m) and comparison to filtration theory for cake filtration (a, solid line) with constant solid volume fraction ($\phi_{s,c} = 0.1$). The initial slow increase of P (a, circles) is attributed to the effect of capillary plugging (b). Once plugging has taken place ($t > 630$ s) the cake builds up with constant solid volume fraction as predicted by cake filtration theory (a, solid line).

The shape of the pressure-drop curve is compared to the one obtained

from filtration theory (eq. (14)) at a constant solid volume fraction of 0.1 (fig. 5.12, Pe = 1). After the capillary has been plugged (630 s), the pressure-drop evolution (fig. 5.13a) predicted by the LD model asymptotically approaches the one predicted from filtration theory. This is due to the growth of a cake with constant solid volume fraction (figure 5.7), which is the main assumption in classical filtration theory. Before plugging, filtration theory fails to predict the pressure-drop evolution (fig. 5.13a).

Figure 5.14 shows the evolution of the pressure-drop scaled with the fluid velocity U_0 for several Pe numbers (0.01 - 10) as a function of the deposited mass. In contrast to previous filtration studies at Pe $>$ 10 (ballistic limit) [92], the pressure-drop curves do not fall on top of each. The reason for this is the changing solid volume fraction with changing Pe. In fact, the permeability $B_{0,c}$ through the cake decreases by two orders of magnitude when the cake solid volume fraction ($\phi_{s,c}$) increases 0.01 to 0.14 with increasing Pe from 0.01 to 10 (fig. 5.12). The largest increase in pressure-drop is due to the growth of a filter cake, which is the regime predicted by classical filtration theory. Therefore, the accurate prediction of the plugging time and the cake solid volume fraction, obtained in this work, are essential in predicting the pressure-drop evolution by the classical filtration theory.

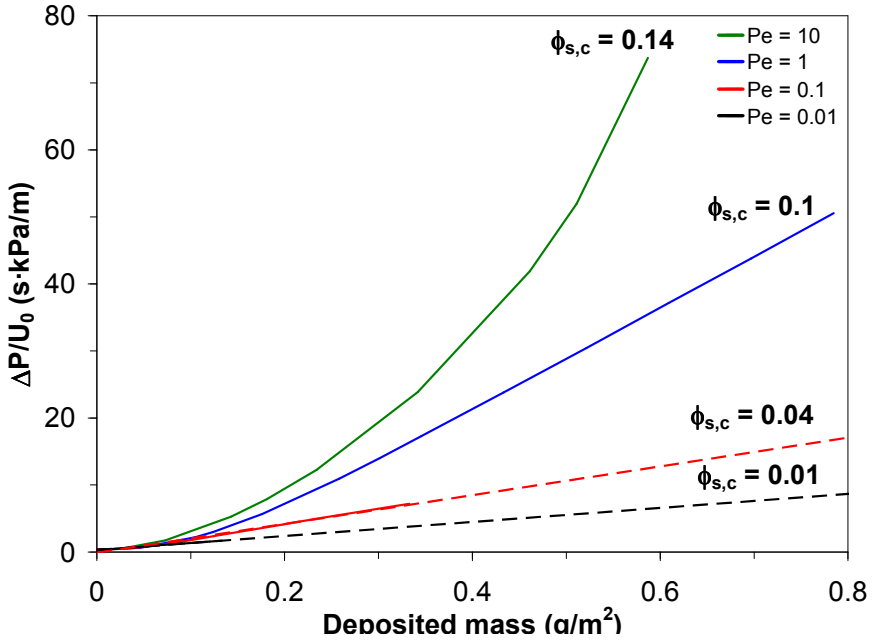


Figure 5.14 Pressure-drop scaled with the flow velocity U0 as a function of the deposited mass. The broken lines (red and black line) are the extrapolation of the pressure-drops by classical cake filtration with constant solid volume fraction. The increase in cake solid volume fraction ($\phi_{s,c}$) from Pe = 0.01 (black broken) to Pe = 10 (green line) decreases the permeability increasing the pressure-drop.

5.4 Comparison to experiments

The following section shows a comparison between the characterization of the morphology of experimentally obtained cakes and the ones derived from the LD model. Due to the limitations of the LD model, it was not possible to capture the "heap filtration", that is the transition between plugging of the substrate capillaries and cake filtration. Nevertheless, the plugging times obtained with the LD model are directly comparable with those obtained from experiments, since it is only for times larger than the plugging time, that the difference between model and experiments will show. The constant cake porosity obtained with the LD model will be compared to the ones obtained from experiments, even though the Pe number changed during the experiments since the pressure-drop was kept constant. This is possible, since the flow-rate did not decrease to more than 80 - 70% of the initial flow-rate. The Pe number can therefore be assumed approximately constant throughout filtration.

5.4.1 Model parameters

Table 5.2 shows the parameters of the LD model compared to typical experimental ones.

Table 5.2 Parameters in the LD simulations compared to typical experimental ones.

Parameter	Model	Experimental	Unit
Peclet number, Pe	0.1	0.3 - 0.5$^{(1)}$	-
Cap. radius, R_c	2	2$^{(2)}$	μm
Particle (sphere) size, d_p	25	27.9$^{(3)}$	nm
Number concentration	$7.38 \cdot 10^{14}$	$7.38 \cdot 10^{14}$	#/m^3

$^{(1)}$: See table 3.4
$^{(2)}$: See section 3.3
$^{(3)}$: See section 2.4

Due to model and time restrictions, particles were assumed to be spherical and monodisperse, even though SMPS found them to be agglomerated with an average agglomerate size of 27.9 nm (cf. section 2.4). Agglomerated particles should yield larger porosities for all Pe numbers, as a result of their earlier interception [10] (larger collision radius). It is possible to correct for the existence of fractal-like agglomerates, however it requires a different method of calculating collisions than the one presented in section 5.2.3. Also a different equation is required for the calculation of the friction factor, which is used in the integration of the Langevin equation of motion (eq. 5.1). This can possibly be done as suggested by Heine et al. [98], by using an effective collision diameter from equation 2.1 assuming a fractal

dimension of the agglomerates. However, it was not possible to correct for agglomerated particles within the time frame of this project.

Using the size determined by SMPS, the number concentration was calculated from the known mass-concentration ($3.27 \cdot 10^{-5}$ kg/m^3, see table 2.4). It should be mentioned, that as the fractal dimension of the agglomerates was not known, the bulk density was used in the calculations.

The chosen Pe number was slightly smaller than typical experimental ones (0.1 compared to 0.3 - 0.5). However, the choice of values in table 5.2 are still comparable to the experimental conditions.

5.4.2 Comparison to SEM

Figure 5.15 shows an SEM (a) of alumina agglomerates with a average agglomerate size of 27.9 nm deposited on an α-alumina substrate (experiment label: jta-perm-18, see table 3.4), compared to the shape from the LD model (b) obtained structure assuming 25 nm particles (deposited to a height of 9 μm at Pe = 0.1).

As expected, from the deposition of 50 nm particles, little penetration is seen (fig. 5.15b) in the capillary and the interface between deposit and capillary is clear. Compared to SEM (fig. 5.15a), this is reasonable as the interface between deposit and capillary seems equally clear. Such an interface is likely to yield poor adhesion to the substrate, since the particle cake is only anchored at a few points to the surface.

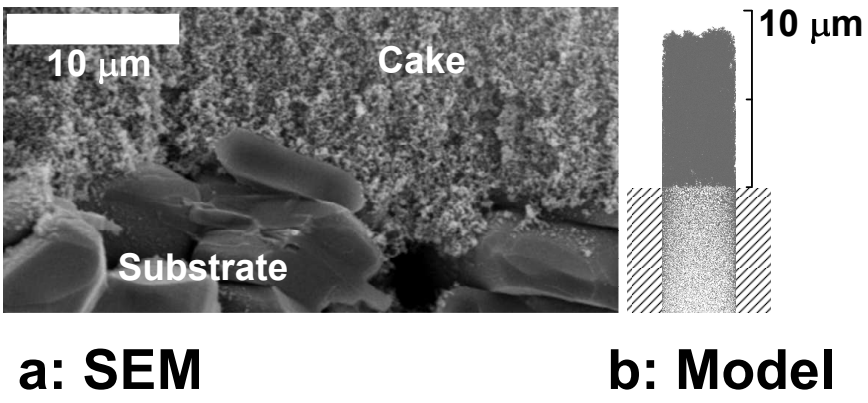


Figure 5.15 SEM (a) of jta-perm-18 compared to the LD model (b) for deposition of 25 nm spherical particles at Pe = 0.1 and $R_c = 2$ μm, showing the substrate-cake interface. The scale applies for both pictures. The deposition conditions in the model are comparable to the experimental values (see table 5.2).

5.4.3 Comparison to filtration curves

Figure 5.16 shows the fraction of particles entering through the capillary mouth as a function of time. The three filtration identified earlier in figure 5.8 are seen. However, the regimes are different from those identified by the experimentally obtained filtration curves: whereas capillary deposition is seen in the model and not in experiments, heap filtration is seen in the experiments, and not in the model. Pure capillary deposition is finished after only a few seconds, which explains why it is not seen experimentally. The plugging time is found to be ≈ 300 seconds (5 min) and the (constant) porosity is obtained from the solid volume fraction profile ($\epsilon_c = 1 - \phi_{s,c}$) (not shown).

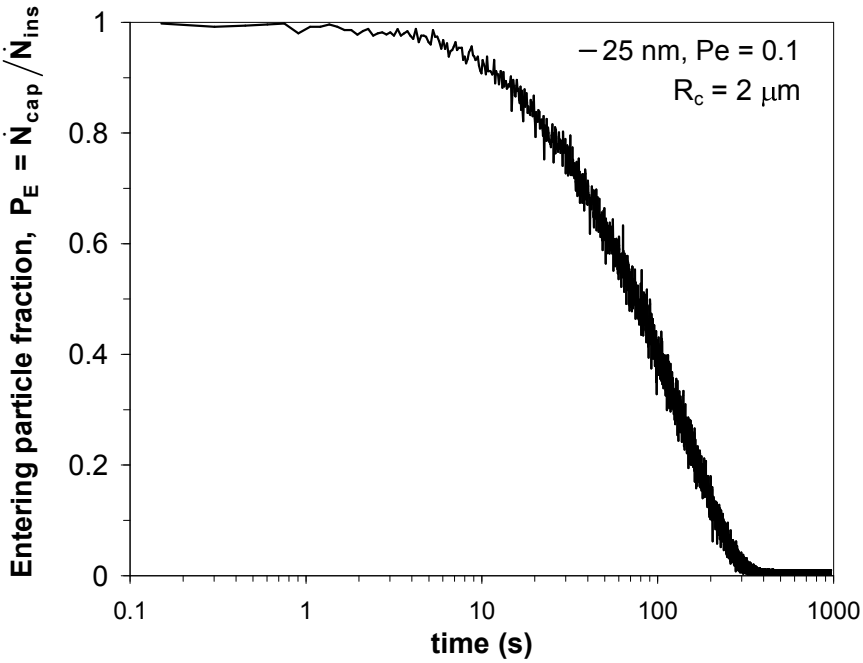


Figure 5.16 Fraction of particles ($d_p = 25$ nm) entering through the capillary mouth ($R_c = 2$ μm) at Pe = 0.1. Capillary plugging occurs after approximately 300 seconds (5 min). Number concentration: $C_n = 7.38 \cdot 10^{14}$ #/m^3.

5.4.4 Comparison summary

5.4.5 Discussion

Table 5.3 shows a comparison of the plugging time and cake porosity obtained by the LD model and averaged experimental values.

In general, good comparison between experimentally obtained values and those of the simulation are found (table 5.3). The LD simulations predict a somewhat higher plugging time than observed experimentally, which is

Table 5.3 Comparison of LD model to experiments with similar deposition conditions.

$\epsilon_{c,LD}$	$\epsilon_{c,exp}^{(1)}$	$t_{plug,LD}$	$t_{plug,exp}^{(2)}$
-	-	minutes	minutes
0.953	0.94 - 0.97	5	1.7

(1): From SEM, see section 3.4.1

(2): Determined from filtration curves as the time at which the "knee" begins.

most likely due to the lower Pe number, which yields a lower particle flux than in the experiments. Furthermore, in experiments, particles are not monodisperse and some large agglomerates may exists, which can "lodge" themselves in the capillary and shade off the capillary faster than with only monodisperse particles.

The constant porosity obtained from the solid volume fraction profile (0.953) is comparable to the experimentally obtained one (0.94 - 0.97). Interestingly, the constant porosity obtained for 25 nm particles at Pe = 0.1 (0.953) is also smaller than for 50 nm particles at Pe = 0.1 (0.96, cf. figure 5.12). Mädler et al. [51] already showed that this is due the larger free mean path of the smaller particles, leading to a later interception and lower deposit porosity than for larger particles.

The *average* porosity is calculated outside of the capillary, since the porosity inside the capillary is approximately equal to 1:

$$\epsilon_{c,avr} = \frac{\int_{z=0}^{z'}(1-\phi_s(z))dz}{z'} \tag{5.33}$$

where $\phi_s(z)$ is the solid volume fraction profile through the cake deposited outside of the capillary. The deposited mass per substrate capillary is known from the LD model. The total mass deposited can be estimated by assuming that particles only deposit on top of the substrate capillaries. As expected, this assumption becomes worse the thicker the cake is grown. The number of substrate capillaries ($n_{c,s}$) can be found from a balance on the substrate capillary volume:

$$n_{c,s} = \frac{\epsilon_s A_c}{\frac{\pi}{4} d_{cs}^2} \tag{5.34}$$

where A_c is the filtration area. Figure 5.17 shows the average porosity (outside of capillary) as a function of the deposited mass per capillary as well as the total mass calculated from equation 5.34. The average porosity approaches an asymptotic value (the cake constant porosity) with increasing deposited mass. The figure shows, that an average porosity of 0.96, which is considerably less than 1, should be obtained already before 0.7 mg has been deposited, which was the lowest mass deposited in the experiments.

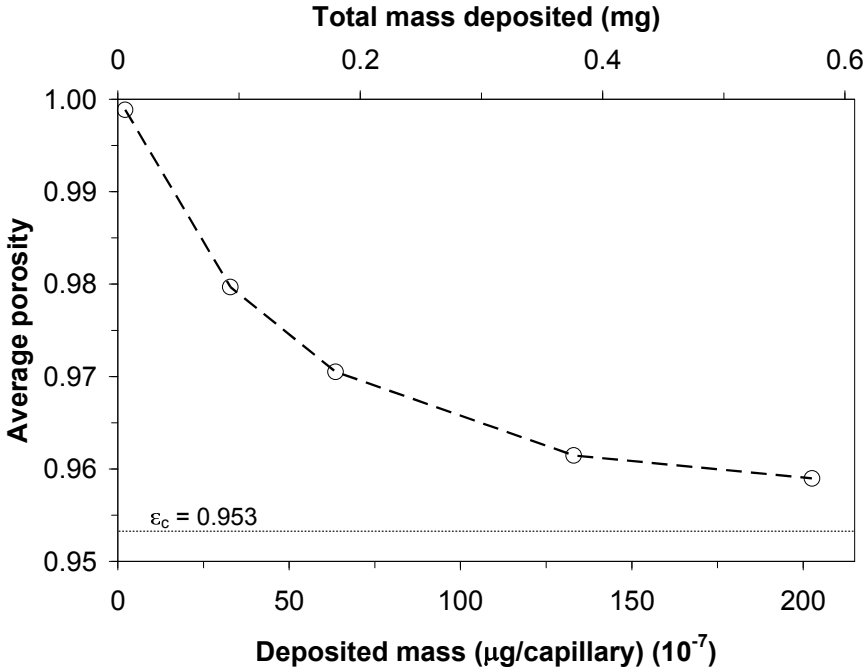


Figure 5.17 Average porosity of deposit of 25 nm particles at Pe = 0.1 as a function of deposited mass per substrate capillary (lower abscissa) and total deposited mass (upper abscissa). The total deposited mass has been calculated using the assumptions stated in the text. The porosity was calculated by applying eq. 5.33 solid volume fraction profile. The porosity converges toward a constant value for increasing deposited mass. A large difference between initial value and end value can be seen.

5.4.6 Possible application of the filtration method

One of the possible and attractive applications of the highly porous structures generated with the filtration method studied in this work, is the deposition of catalytic particles for soot oxidation on a particle filter. The high porosity of the cakes, combined with the high filtration velocity during particulate filtration, allows the soot particles to penetrate into the cake, where they will be trapped and combusted. As they are combusted and shrink they may become detached and continue their path down through the cake. Because of the complex fractal-like network that the cake is, diesel particles combusting may move down through the cake touching more cake particles on their way. The required residence time for complete combustion can be adjusted by controlling the cake thickness. The possible mechanism is showed schematically on figure 5.18.

The previous scenario only works if the cake porosity is so high, that significant penetration can occur through the cake capillaries. Another important point recently backed up by TEM studies [21], is a good contact between the soot and the catalytically active particles, also called "tight contact" [18]. If the the diesel particles do not penetrate, a soot cake will form on top of the other cake and all the catalytic activity of the particles in the lower cake will be lost.

A cake porosity of 99% may be obtained by deposition of 50 nm particles at Pe = 0.01, as it was shown in figure 5.11. Based on the cylindrical capillary model, this results in a capillary size of 3.3 μm. This seems a little high, however the capillary size may definitely well be in the micron size when considering the structures grown on figure 5.10 at low Pe. An advantage of the high porosity is also an obvious low pressure-drop across the cake, even at high fluid velocities during diesel particle filtration.

5.4.7 Link to computer model

Further work includes the extended simulation of the penetration of soot nanoparticles at high Pe into a cake which has been grown at low Pe.

The following strategy is proposed for this purpose:

1. Deposition of a cake of nanoparticles at a given Pe.
2. Increase of the particle size to the soot particle size ($>$ 100 nm) and increase the Pe number due to both lower diffusion and higher fluid velocity.
3. Apply a "perfect sink" boundary condition, so that the soot particles do not modify the structure. This corresponds to a situation where soot particles instantaneously combust upon touching the catalytically active particles.

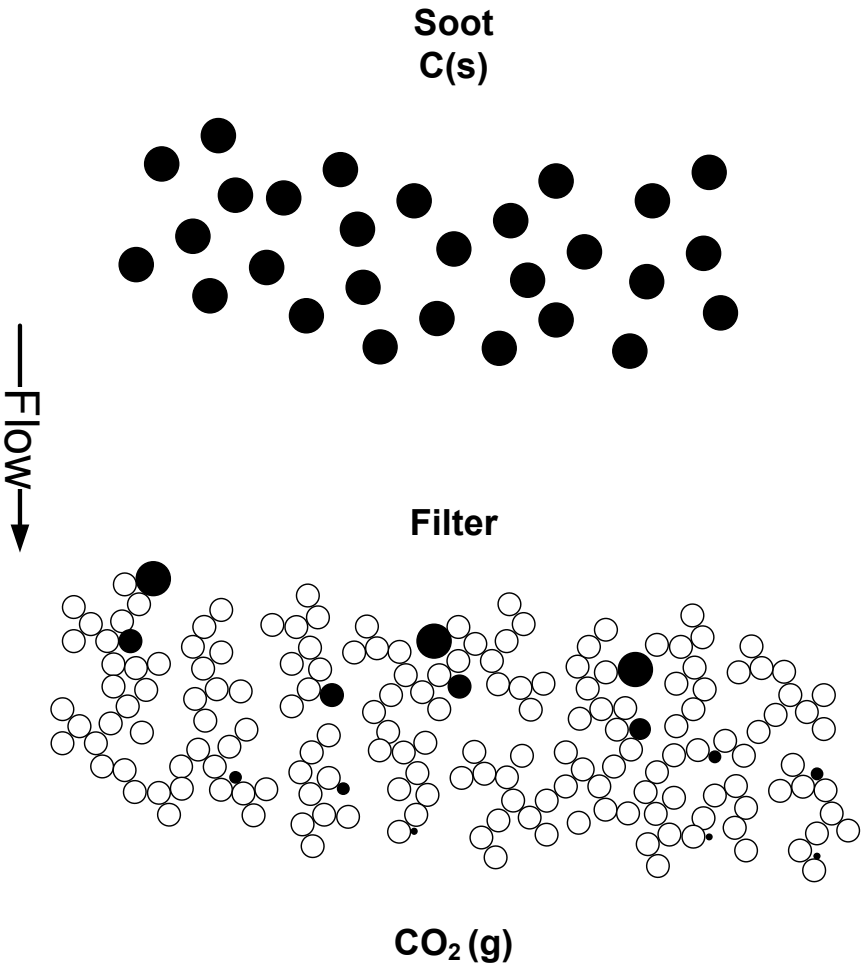


Figure 5.18 Possible mechanism for soot capture in highly porous ceramic cakes consisting of catalytically active nanoparticles. The cakes can be formed at by filtration at low Pe number with up to 99% porosity. Soot is caught in the structure and combusted. As the particle combusts it shrinks and continues to pass through the catalytically active cake. Finally, pure CO_2 exists the filter.

With this method it will be possible to investigate if a high penetration of diesel particles can be obtained and to predict optimal process conditions. It requires some modification of the computer code, to allow the deposition of polydisperse particles.

5.5 Conclusions

The full 3D morphology and time-evolution of nanoparticle deposits during filtration as a function of the Pe number (0.01 - 10), was obtained by langevin dynamics (LD). The transition between capillary and cake filtration was captured by allowing the particle deposits to grow out of the capillary. Three distinct time-regimes were observed. In the first regime (capillary deposition), deposition occurred inside the capillary with the penetration profile being well represented by the perfect sink model. In the second regime (capillary plugging), deposition occurred both inside and outside of the capillary, effectively shading off the capillary mouth. This changed the particle penetration profile inside the capillary and the perfect sink model became inaccurate once significant shading had taken place. In the last regime (cake growth), the capillary was fully plugged as no particle penetration occurred through the capillary mouth. As a consequence, depositing particles formed a cake with constant solid volume fraction ($\phi_{s,c}$).

Plugging was observed to take place mainly outside of the capillary as 83 - 60 % (Pe = 0.01 - 10) of the total particle mass was deposited there at the time of capillary plugging. The time of capillary plugging (t_{plug}), which marked the onset of cake growth, was found by monitoring the particle penetration through the capillary mouth as a function of the Pe number. This time decreased from 10000 s to 130 s with Pe increasing from 0.01 to 10. The plugging time was not linearly decreasing with increasing flux, as the higher Pe increased the solid volume fraction of the cake, forcing more particles to deposit until the time of plugging.

At low Pe the structure was fractal-like with a low cake solid volume fraction ($\phi_{s,c}$ = 0.01 at Pe = 0.01). At high Pe, the structure converged toward the ballistic determined morphology with a significantly higher cake solid volume fraction ($\phi_{s,c}$ = 0.14 at Pe = 10). The plugging height increased from 3.0 to 5.7 μm with Pe increasing from 0.01 to 10. At higher Pe, the formation of a void cone at was observed with the tip of the cone corresponded well to the plugging height.

The pressure-drop time-evolution for constant flow filtration was modeled. The shape of the pressure-drop evolution curves qualitatively described the experimentally obtained ones. Initially the pressure-drop was low and the increase was slow as particles mainly deposited in the capillary. As the capillary mouth became increasingly shaded off, the pressure drop started increasing more rapidly. Therefore, the pressure-drop rise due to capillary

deposition was negligible for all Pe numbers. After capillary plugging, the pressure-drop evolution asymptotically reached the classical theory for cake filtration with constant solid volume fraction $\phi_{s,c}$.

The cake permeability was decreased by two orders of magnitude with increasing Pe from 0.01 to 10. This increased the pressure-drop above what was expected by mainly increasing the flow. As a significant part of the pressure-drop increase occurred during cake growth, the plugging time and the cake solid volume fraction obtained in this work are two fundamental parameters for the use of classical filtration theory.

Good comparison between experimental work (at Pe = 0.3 - 0.5) and the simulated deposition by filtration of 25 nm spherical particles at Pe = 0.1 was obtained. The plugging time calculated from the LD simulations, was found to be approximately 5 minutes, while for the experiments, an average plugging time of 1.7 minutes was found. The main reason for this difference is the lower particle flux in the simulations. The LD calculated constant porosity (0.953) was in good comparison to the experimentally calculated porosity (0.94 - 0.97 from SEM). The average porosity was seen to slowly decrease toward the constant value. For a deposited mass of 0.7 mg, which was the lowest mass deposited in the experiments, the model showed that the porosity should be substantially less than 1 (0.96 after 0.6 mg).

Finally, it was proposed that a modified version of the model could be used for in order to study the deposition of soot in a filter medium. By doing so, optimal morphology of the filter, as well as process parameters for a good soot-catalyst contact, may be found.

Chapter 6

Conclusions and outlook

The deposition of flame-made nanoparticles by filtration on porous α-alumina substrates, was studied in detail by experiments, as well as through a novel modelling approach. The deposited cakes were made from agglomerates of alumina nanoparticles, with a mean agglomerate diameter of 27.9 nm. Three regimes of deposition could be observed from the filtration curves obtained for constant pressure-drop filtration: capillary plugging, transition ("heap filtration") and finally cake growth. Capillary deposition was not observed in the experiments, but was shown by modelling to end after only a few seconds from the onset of filtration.

The cake (average) capillary size was 160 - 227 nm and the calculated porosities were in the range of 0.94 - 0.97 (SEM) regardless of the deposited mass. The porosity calculated from permeability measurements decreased with increasing mass, and best comparison to SEM was obtained for cakes of more than 8 mg deposited mass (0.97). Both characterization methods showed that a high porosity of "randomly packed" particles was detrimental to a small capillary size. In the investigated range of Pe numbers (0.3 - 0.5), no difference was seen in the cake capillary size or the calculated porosities. If the application of the porous deposits require a smaller capillary size (such as in membrane separation), a higher Pe number is needed which lowers the porosity of the cake and in turn also the capillary size of the cake. However, it is not possible by the filtration method to obtain porosities lower than the ballistically limited ones (0.85) as confirmed by modelling. Restructuring is required if lower porosity is desired.

Thick (≈ 28 μm) as-deposited cakes were overall intact after sintering for 2 hours up to 900 °C with a moderate increase in the cake capillary size. At 1100 °C the cakes were completely disintegrated due to a large free shrinkage. Thin (≈ 3 μm) cakes showed a drastic increase in the cake capillary size already at 900 °C. The reason for the difference between thick and thin cakes, was explained by the observation of micro-cracks, which for thinner cakes affected the permeability more significantly than for thicker ones. The exact

mechanism for the formation of these cracks is not fully understood, however it was proposed that they may occur from the compression/contraction of the cake in the substrate capillaries during the heating/cooling cycle.

The mechanical stability of thick as-deposited cakes was improved by sintering for 2 hours at 700 °C. Whereas, as-deposited cakes were destroyed at high relative saturation pressures of cyclohexane ($S = 0.9 - 1$) due to condensation in the cake capillaries, sintered cakes remained intact. This is most likely because the as-deposited cakes consisted of particles bound together by weak Van der Waals forces, which were smaller than the capillary forces at high relative saturation pressures ($S > 0.9$). In contrast, by heating the cakes to 700 °C, the formation of strong sintering necks prevented this destruction.

The deposition and pressure-drop evolution of particles during filtration was modelled using Langevin dynamics for several Pe numbers (0.01 - 10). For all Pe numbers, plugging took place on the outside of the substrate capillary, which was in good comparison to the experimental results. In fact, approximately 60% of the deposited mass was deposited on the outside of the substrate capillary at the time of plugging (Pe = 10). From the penetration curves three regimes could be observed: Capillary deposition, capillary plugging and cake growth. The "heap filtration" transition regime observed experimentally between capillary plugging and cake growth could not be modelled with the chosen geometry. The structure of the deposits were fractal-like at low Pe, while at higher Pe, the structures resembled that of a void cone with the plugging height placed at the cone tip. Once the capillary was plugged, the solid volume fraction profile (1-porosity) reached a constant level. At Pe = 0.01, the constant solid volume fraction was 0.01, while at Pe = 10, this value was 0.14, which was in good comparison to that which was obtained in literature for flat substrate deposition. Also, good agreement of the model to the experiments of this work was seen for deposition of 25 nm particles at Pe = 0.1: The plugging time was 5 minutes compared to experimental 1.7 minutes, and the constant porosity was 0.95 compared to experimental 0.94 - 0.98.

6.1 Research suggestions

This work has shown, that it was not possible without restructuring to deposit cakes with sufficiently small capillaries for the use in e.g. membrane separation. However, it has been found that highly porous structures can be formed which may be interesting for soot oxidation in particle filters of diesel-driven cars. Future work should therefore concentrate on the application of the cakes within this field. This can be done experimentally by depositing catalytically active particles at low Pe number (≈ 0.01). A numerical study on the penetration of soot particles through the deposited

cakes may be carried out by modifying the deposition model developed in this work. A high penetration is important, not only to achieve a good soot-catalyst contact, but also to use as much of the filter thickness as possible for deep bed filtration, as a cake of uncombusted soot may otherwise build up and plug the filter.

Another interesting finding is that, although capillary plugging occurs fast, it is still possible to deposit particles directly within the porous substrate. This may be of interest for any sort of surface coating with functional nanoparticles. The process may be optimized with the model, as it is possible to examine, for several different deposition conditions, at which point capillary deposition ends and capillary plugging begins.

In this work, the Pe number was only varied over a small range and in future studies it would be interesting to extend this range to > 10, as the ballistic limit of porosity is reached here. This can be done by changing the substrate, to allow for larger gas velocities. Alternatively, an over-pressure deposition cell can be constructed which can allow a larger pressure difference between the high and low pressure side.

Future work on the model should include a more realistic flow-profile in the capillary, as well as in the cake domain. Furthermore, the simultation of deposition of agglomerated particles during filtration should be carried out, as deposition of these have shown to affect the final cake porosity.

Bibliography

[1] R. Strobel, A. Baiker, and S. E. Pratsinis. Aerosol flame synthesis of catalysts. *Advanced Powder Technology*, 17(5):457–480, 2006.

[2] G. L. Chiarello, J. D. Grunwaldt, D. Ferri, R. Krumeich, C. Oliva, L. Forni, and A. Baiker. Flame-synthesized $LaCoO_3$-supported Pd 1. structure, thermal stability and reducibility. *Journal of Catalysis*, 252(2):127–136, 2007.

[3] R. Strobel and S. E. Pratsinis. Flame aerosol synthesis of smart nanostructured materials. *Journal of Materials Chemistry*, 17(45):4743–4756, 2007.

[4] S. E. Pratsinis. Flame aerosol synthesis of ceramic powders. *Progress in Energy and Combustion Science*, 24(3):197–219, 1998.

[5] T. Johannessen, J. R. Jensen, M. Mosleh, J. Johansen, U. Quaade, and H. Livbjerg. Flame synthesis of nanoparticles - applications in catalysis and product/process engineering. *Chemical Engineering Research and Design*, 82(A11):1444–1452, 2004.

[6] K. Wegner and S. E. Pratsinis. Gas-phase synthesis of nanoparticles: scale-up and design of flame reactors. *Powder Technology*, 150(2):117–122, 2005.

[7] J. R. Jensen, T. Johannessen, S. Wedel, and H. Livbjerg. Preparation of ZnO-Al_2O_3 particles in a premixed flame. *Journal of Nanoparticle Research*, 2(4):363–373, 2000.

[8] J. R. Jensen, T. Johannessen, S. Wedel, and H. Livbjerg. A study of Cu/ZnO/Al_2O_3 methanol catalysts prepared by flame combustion synthesis. *Journal of Catalysis*, 218(1):67–77, 2003.

[9] R. Mueller, L. Mädler, and S. E. Pratsinis. Nanoparticle synthesis at high production rates by flame spray pyrolysis. *Chemical Engineering Science*, 58(10):1969–1976, 2003.

[10] L. Mädler, A. Roessler, S. E. Pratsinis, T. Sahm, A. Gurlo, N. Barsan, and U. Weimar. Direct formation of highly porous gas-sensing films by in situ thermophoretic deposition of flame-made Pt/SnO_2 nanoparticles. *Sensors and Actuators B-Chemical*, 114(1):283–295, 2006.

[11] A. Tricoli, M. Graf, S. Kühne, Mayer F., A. Hierlemann, and S. E. Pratsinis. Micropatterning layers by flame aerosol deposition-annealing. *Advanced Materials*, Accepted, 2008, 2008.

[12] B. Schimmöller, H. Schulz, S. E. Pratsinis, A. Bareiss, A. Reitzmann, and B. Kraushaar-Czarnetzki. Ceramic foams directly-coated with flame-made V_2O_5/TiO_2 for synthesis of phthalic anhydride. *Journal of Catalysis*, 243(1):82–92, 2006.

[13] S. Thybo, S. Jensen, J. Johansen, T. Johannessen, O. Hansen, and U. J. Quaade. Flame spray deposition of porous catalysts on surfaces and in microsystems. *Journal of Catalysis*, 223(2):271–277, 2004.

[14] D. Chakraborty, H. Bischoff, I. Chorkendorff, and T. Johannessen. Mixed phase Pt-Ru catalyst for direct methanol fuel cell anode by flame aerosol synthesis. *Journal of the Electrochemical Society*, 152(12):A2357–A2363, 2005.

[15] S. K. Andersen, T. Johannessen, M. Mosleh, S. Wedel, J. Tranto, and H. Livbjerg. The formation of porous membranes by filtration of aerosol nanoparticles. *Journal of Nanoparticle Research*, 4(5):405–416, 2002.

[16] K. Hinot, H. Burtscher, A. P. Weber, and G. Kasper. The effect of the contact between platinum and soot particles on the catalytic oxidation of soot deposits on a diesel particle filter. *Applied Catalysis B-Environmental*, 71(3-4):271–278, 2007.

[17] A. P. Walker. Controlling particulate emissions from diesel vehicles. *Topics in Catalysis*, 28(1-4):165–170, 2004.

[18] B.A.A.L. van Setten, M. Makkee, and J.A. Moulijn. Science and technology of catalytic diesel particulate filters. *Catalysis Reviews*, 43(4):489–564, 2001.

[19] M. Ambrogio, G. Saracco, and V. Specchia. Combining filtration and catalytic combustion in particulate traps for diesel exhaust treatment. *Chemical Engineering Science*, 56(4):1613–1621, 2001.

[20] E. Cauda, D. Mescia, D. Fino, G. Saracco, and V. Specchia. Diesel particulate filtration and combustion in a wall-flow trap hosting a $LiCrO_2$ catalyst. *Industrial and Engineering Chemistry Research*, 44(25):9549–9555, 2005.

[21] S. B. Simonsen, S. Dahl, E. Johnson, and S. Helveg. Ceria-catalyzed soot oxidation studied by environmental transmission electron microscopy. *Journal of Catalysis*, 255(1):1–5, 2008.

[22] J. Johansen and T.D. Elmøe. Preparation of porous ceramic membranes by deposition of flame synthesized nanoparticles. *Journal of the Danish Ceramic Society*, 9(1):14–17, 2007.

[23] J. Johansen. *Synthesis of ceramic membranes by deposition of aerosol particles*. Phd thesis, Technical University of Denmark, 2006.

[24] M. Mosleh. *Preparation of Micro Porous Ceramic Membranes by Flame Generated Aerosol Nano-Particles*. Phd thesis, Technical University of Denmark, 2005.

[25] A. A. Lall and S. K. Friedlander. On-line measurement of ultrafine aggregate surface area and volume distributions by electrical mobility analysis: 1. theoretical analysis. *Journal of Aerosol Science*, 37(3):260–271, 2006.

[26] K. E. J. Lehtinen, R. S. Windeler, and S. K. Friedlander. A note on the growth of primary particles in agglomerate structures by coalescence. *Journal of Colloid and Interface Science*, 182(2):606–608, 1996.

[27] T. Johannessen, S. E. Pratsinis, and H. Livbjerg. Computational fluid-particle dynamics for the flame synthesis of alumina particles. *Chemical Engineering Science*, 55:177–191, 2000.

[28] S. Tsantilis and S. E. Pratsinis. Soft- and hard-agglomerate aerosols made at high temperatures. *Langmuir*, 20(14):5933–5939, 2004.

[29] Sheldon K. Friedlander. *Smoke, dust, and haze : fundamentals of aerosol dynamics*. Oxford University Press, New York, second edition, 2000.

[30] F. F. Lange. Sinterability of agglomerated powders. *Journal of the American Ceramic Society*, 67(2):83–89, 1984.

[31] M. K. Wu and S. K. Friedlander. Enhanced power-law agglomerate growth in the free-molecule regime. *Journal of Aerosol Science*, 24(3):273–282, 1993.

[32] C.N. Davies. Definitive equations for the fluid resistance of spheres. *Proc. Phys. Soc.*, 54(4):259–270, 1945.

[33] P. Vainshtein and M. Shapiro. Mobility of permeable fractal agglomerates in slip regime. *Journal of Colloid and Interface Science*, 284(2):501–509, 2005.

[34] P. Tandon and D. E. Rosner. Translational brownian diffusion-coefficient of large (multiparticle) suspended aggregates. *Industrial and Engineering Chemistry Research*, 34(10):3265–3277, 1995.

[35] G. M. Wang and C. M. Sorensen. Diffusive mobility of fractal aggregates over the entire knudsen number range. *Physical Review E*, 60(3):3036–3044, 1999.

[36] T. Johannessen. *Synthesis of Nano-Particles in Flames*. PhD thesis, Technical University of Denmark, 1999.

[37] B. Dahneke. Viscous resistance of straight-chain aggregates of uniform spheres. *Aerosol Science and Technology*, 1(2):179–185, 1982.

[38] H. Y. Wen, G. P. Reischl, and G. Kasper. Bipolar diffusion charging of fibrous aerosol-particles .1. charging theory. *Journal of Aerosol Science*, 15(2):89–101, 1984.

[39] William C. Hinds. *Aerosol technology properties, behavior, and measurement of airborne particles*. Wiley, New York, second edition, 1999.

[40] A. Wiedensohler. An approximation of the bipolar charge-distribution for particles in the sub-micron size range. *Journal of Aerosol Science*, 19(3):387–389, 1988.

[41] J. P. Hansen, J. R. Jensen, H. Livbjerg, and T. Johannessen. Synthesis of ZnO particles in a quench-cooled flame reactor. *Aiche Journal*, 47(11):2413–2418, 2001.

[42] A. A. Lall, M. Seipenbusch, W. Z. Rong, and S. K. Friedlander. On-line measurement of ultrafine aggregate surface area and volume distributions by electrical mobility analysis: II. comparison of measurements and theory. *Journal of Aerosol Science*, 37(3):272–282, 2006.

[43] C.N. Davies. Clogging of fibrous aerosol filters. *Journal of Aerosol Science*, 1:35–39, 1970.

[44] H. Leibold and J. G. Wilhelm. Investigations into the penetration and pressure-drop of hepa filter media during loading with submicron particle aerosols at high-concentrations. *Journal of Aerosol Science*, 22:S773–S776, 1991.

[45] J. I. T. Stenhouse and R. Trottier. The loading of fibrous filters with submicron particles. *Journal of Aerosol Science*, 22:S777–S780, 1991.

[46] P. Penicot, D. Thomas, P. Contal, D. Leclerc, and J. Vendel. Clogging of HEPA fibrous filters by solid and liquid aerosol particles: An experimental study. *Filtration and Separation*, 36(2):59–64, 1999.

[47] J. C. Ruiz, P. Blanc, E. Prouzet, P. Coryn, P. Laffont, and A. Larbot. Solid aerosol removal using ceramic filters. *Separation and Purification Technology*, 19(3):221–227, 2000.

[48] K. R. Spurny, J. P. Lodge, E. R. Frank, and D. C. Sheesley. Aerosol filtration by means of nuclepore filters structural and filtration properties. *Environmental Science and Technology*, 3(5):453–464, 1969.

[49] K. R. Spurny, J. Havlova, J. P. Lodge, E. R. Ackerman, D. C. Sheesley, and B. Wilder. Aerosol filtration by means of nuclepore filters - filter pore clogging. *Environmental Science and Technology*, 8(8):758–761, 1974.

[50] T.T. Kodas and Mark Hampden-Smith. *Aerosol Processing of Materials*. Wiley-VCH, first edition, 1999.

[51] L. Mädler, A. A. Lall, and S. K. Friedlander. One-step aerosol synthesis of nanoparticle agglomerate films: simulation of film porosity and thickness. *Nanotechnology*, 17(19):4783–4795, 2006.

[52] M. R. Wiesner. Morphology of particle deposits. *Journal of Environmental Engineering-Asce*, 125(12):1124–1132, 1999.

[53] P. Kulkarni and P. Biswas. A brownian dynamics simulation to predict morphology of nanoparticle deposits in the presence of interparticle interactions. *Aerosol Science and Technology*, 38(6):541–554, 2004.

[54] D. Rodriguez-Perez, J. L. Castillo, and J. C. Antoranz. Relationship between particle deposit characteristics and the mechanism of particle arrival. *Physical Review E*, 72(2):021403–1 – 021403–9, 2005.

[55] J. Blum and R. Schräpler. Structure and mechanical properties of high-porosity macroscopic agglomerates formed by random ballistic deposition. *Physical Review Letters*, 93(11):115503–1 – 115503–4, 2004.

[56] R. Jackson. *Transport in porous catalysts*. Elsevier Scientific Pub. Co., New York, 1977.

[57] S. Thomas, R. Schafer, J. Caro, and A. Seidel-Morgenstern. Investigation of mass transfer through inorganic membranes with several layers. *Catalysis Today*, 67(1-3):205–216, 2001.

[58] Julian Szekely, James W. Evans, and Hong Yong Sohn. *Gas-solid reactions*. Academic Press, New York, 1976.

[59] T. Elias-Kohav, M. Sheintuch, and D. Avnir. Steady-state diffusion and reactions in catalytic fractal porous-media. *Chemical Engineering Science*, 46(11):2787–2798, 1991.

[60] A. Nakayama, F. Kuwahara, and Y. Sano. Concept of equivalent diameter for heat and fluid flow in porous media. *Aiche Journal*, 53(3):732–736, 2007.

[61] N. Epstein. On tortuosity and the tortuosity factor in flow and diffusion through porous-media. *Chemical Engineering Science*, 44(3):777–779, 1989.

[62] S. Ergun and A.A. Orning. Fluid flow through randomly packed columns and fluidized beds. *Industrial and Engineering Chemistry*, 41:1179–1184, 1949.

[63] J.G. Sanches Marcano and T.T. Tsotsis. *Catalytic Membranes and Membrane Reactors*. Wiley-VCH Verlag GmbH, Weinheim, first edition, 2002.

[64] P. Kulkarni and P. Biswas. Morphology of nanostructured films for environmental applications: Simulation of simultaneous sintering and growth. *Journal of Nanoparticle Research*, 5(3-4):259–268, 2003.

[65] Y. Liu, E. Koep, and M. L. Liu. Highly sensitive and fast-responding SnO_2 sensor fabricated by combustion chemical vapor deposition. *Chemistry of Materials*, 17(15):3997–4000, 2005.

[66] J. N. Calata, G. Q. Lu, and T. J. Chuang. Constrained sintering of glass, glass-ceramic and ceramic coatings on metal substrates. *Surface and Interface Analysis*, 31(7):673–681, 2001.

[67] D. R. Carroll and M. N. Rahaman. An initial-stage model for the sintering of constrained polycrystalline thin-films. *Journal of the European Ceramic Society*, 14(5):473–479, 1994.

[68] T. J. Garino and H. K. Bowen. Deposition and sintering of particle films on a rigid substrate. *Journal of the American Ceramic Society*, 70(11):C315–C317, 1987.

[69] S. Bertaux, P. Reynders, and J. M. Heintz. Sintering of nanocrystalline Ta_2O_5 and ZrO_2 films compared to that of TiO_2 films. *Journal of the European Ceramic Society*, 26(6):923–932, 2006.

[70] D. J. Chen and M. J. Mayo. Densification and grain growth of ultrafine 3 mol ceramics. *NanoSTRUCTURED MATERIALS*, 2:469–478, 1993.

[71] K. Ada, M. Önal, and Y. Sarikaya. Investigation of the intra-particle sintering kinetics of a mainly agglomerated alumina powder by using surface area reduction. *Powder Technology*, 168(1):37–41, 2006.

[72] A. Kobata, K. Kusakabe, and S. Morooka. Growth and transformation of TiO_2 crystallites in aerosol reactor. *Aiche Journal*, 37(3):347–359, 1991.

[73] H. Schaper, E. B. M. Doesburg, and L. L. Vanreijen. The influence of lanthanum oxide on the thermal-stability of gamma alumina catalyst supports. *Applied Catalysis*, 7(2):211–220, 1983.

[74] M. J. Mayo, D. C. Hague, and D. J. Chen. Processing nanocrystalline ceramics for applications in superplasticity. *Materials Science and Engineering a-Structural Materials Properties Microstructure and Processing*, 166(1-2):145–159, 1993.

[75] M. N. A. Karlsson, K. Deppert, L. S. Karlsson, M. H. Magnusson, J. O. Malm, and N. S. Srinivasan. Compaction of agglomerates of aerosol nanoparticles: A compilation of experimental data. *Journal of Nanoparticle Research*, 7(1):43–49, 2005.

[76] B. B. Panigrahi and M. M. Godkhindi. Grain growth in ultrafine titanium powders during sintering. *Journal of Nanoparticle Research*, 8(5):627–633, 2006.

[77] R. K. Bordia and G. W. Scherer. On constrained sintering .1. constitutive model for a sintering body. *Acta Metallurgica*, 36(9):2393–2397, 1988.

[78] G. W. Scherer and T. Garino. Viscous sintering on a rigid substrate. *Journal of the American Ceramic Society*, 68(4):216–220, 1985.

[79] J. M. Zheng and J. S. Reed. Effects of particle packing characteristics on solid-state sintering. *Journal of the American Ceramic Society*, 72(5):810–817, 1989.

[80] O. Ogunsola and S. Ehrmann. A monte carlo and continuum study of mechanical properties of nanoparticle based films. *Journal of Nanoparticle Research*, 10:31–39, 2008.

[81] Y. R. Uhm, G. H. Lee, J. H. Park, W. W. Kim, and C. K. Rhee. Study of phase transformation of nano Al_2O_3 compacts derived by hydrolysis and subsequent thermal sintering of Al powders. *Designing, Processing and Properties of Advanced Engineering Materials, Pts 1 and 2*, 449-4:1129–1132, 2004.

[82] R. Strobel, S. E. Pratsinis, and A. Baiker. Flame-made $Pd/La_2O_3/Al_2O_3$ nanoparticles: thermal stability and catalytic behavior in methane combustion. *Journal of Materials Chemistry*, 15(5):605–610, 2005.

[83] H. Noda, K. Muramoto, and H. Kim. Preparation of nano-structured ceramics using nanosized Al_2O_3 particles. *Journal of Materials Science*, 38(9):2043–2047, 2003.

[84] Karl Goossens. Flame synthesis and deposition of Al_2O_3 nanoparticles. investigation of thermal stability, 2007. ETH internal report.

[85] Y.V. Naidich, I.A. Lavrinenko, and V.Y. Petrishchev. Study on the capillary adhesive forces between solid particles with a liquid layer at the points of contact. *Powder Metallurgy and Metal Ceramics*, 4(2):129–133, 1965.

[86] L. T. Canham, A. G. Cullis, C. Pickering, O. D. Dosser, T. I. Cox, and T. P. Lynch. Luminescent anodized silicon aerocrystal networks prepared by supercritical drying. *Nature*, 368(6467):133–135, 1994.

[87] F. P. Cuperus, D. Bargeman, and C. A. Smolders. Characterization of anisotropic uf-membranes - top layer thickness and pore structure. *Journal of Membrane Science*, 61:73–83, 1991.

[88] G. Z. Cao, J. Meijerink, H. W. Brinkman, and A. J. Burggraaf. Permporometry study on the size distribution of active pores in porous ceramic membranes. *Journal of Membrane Science*, 83(2):221–235, 1993.

[89] S. Veerapaneni and M. R. Wiesner. Particle deposition on an infinitely permeable surface - dependence of deposit morphology on particle-size. *Journal of Colloid and Interface Science*, 162(1):110–122, 1994.

[90] F. D. A. A. Reis. Statistical models of diffusion and aggregation for coke formation in a catalyst pore. *Physica a-Statistical Mechanics and Its Applications*, 350(2-4):407–417, 2005.

[91] S. Calle, P. Contal, D. Thomas, D. Bemer, and D. Leclerc. Description of the clogging and cleaning cycles of filter media. *Powder Technology*, 123(1):40–52, 2002.

[92] D. Thomas, P. Penicot, P. Contal, D. Leclerc, and J. Vendel. Clogging of fibrous filters by solid aerosol particles - experimental and modelling study. *Chemical Engineering Science*, 56(11):3549–3561, 2001.

[93] D. A. Japuntich, J. I. T. Stenhouse, and B. Y. H. Liu. Experimental results of solid monodisperse particle clogging of fibrous filters. *Journal of Aerosol Science*, 25(2):385–393, 1994.

[94] V. A. Kirsch. Method for the calculation of an increase in the pressure drop in an aerosol filter on clogging with solid particles. *Colloid Journal*, 60(4):439–443, 1998.

[95] D. Thomas, P. Contal, V. Renaudin, P. Penicot, D. Leclerc, and J. Vendel. Modelling pressure drop in hepa filters during dynamic filtration. *Journal of Aerosol Science*, 30(2):235–246, 1999.

[96] D. L. Ermak and H. Buckholz. Numerical-integration of the langevin equation - monte-carlo simulation. *Journal of Computational Physics*, 35(2):169–182, 1980.

[97] A. Gutsch, S. E. Pratsinis, and F. Löffler. Agglomerate structure and growth-rate by trajectory calculations of monomer-cluster collisions. *Journal of Aerosol Science*, 26(2):187–199, 1995.

[98] M. C. Heine and S. E. Pratsinis. Brownian coagulation at high concentration. *Langmuir*, 23(19):9882–9890, 2007.

[99] M. Tassopoulos, J. A. Obrien, and D. E. Rosner. Simulation of microstructure mechanism relationships in particle deposition. *Aiche Journal*, 35(6):967–980, 1989.

[100] P. Meakin. Formation of fractal clusters and networks by irreversible diffusion-limited aggregation. *Physical Review Letters*, 51(13):1119–1122, 1983.

[101] K. C. Fan and J. W. Gentry. Clogging in nuclepore filters. *Environmental Science and Technology*, 12(12):1289–1294, 1978.

[102] *Ullmann's Encyclopedia of industrial chemistry*. Wiley-VCH, Weinheim, Germany, 7th (online) edition, 2008.

[103] P. Meakin. Diffusion-controlled deposition on surfaces - cluster-size distribution, interface exponents, and other properties. *Physical Review B*, 30(8):4207–4214, 1984.

[104] R.B. Bird, W.E. Stewart, and E.N. Lightfoot. *Transport Phenomena*. John Wiley and Sons Inc., New York, second edition, 2002.

[105] B. Nait-Ali, K. Haberko, H. Vesteghem, J. Absi, and D. S. Smith. Thermal conductivity of highly porous zirconia. *Journal of the European Ceramic Society*, 26(16):3567–3574, 2006.

[106] S. Whitaker. Forced convection heat-transfer correlations for flow in pipes, past flat plates, single cylinders, single spheres, and for flow in packed-beds and tube bundles. *Aiche Journal*, 18(2):361–371, 1972.

[107] L. Mädler, H. K. Kammler, R. Mueller, and S. E. Pratsinis. Controlled synthesis of nanostructured particles by flame spray pyrolysis. *Journal of Aerosol Science*, 33(2):369–389, 2002.

[108] M. C. Heine, L. Mädler, R. Jossen, and S. E. Pratsinis. Direct measurement of entrainment during nanoparticle synthesis in spray flames. *Combustion and Flame*, 144(4):809–820, 2006.

[109] M. Götzinger and W. Peukert. Particle adhesion force distributions on rough surfaces. *Langmuir*, 20(13):5298–5303, 2004.

[110] Q. Li, V. Rudolph, and W. Peukert. London-van der waals adhesiveness of rough particles. *Powder Technology*, 161(3):248–255, 2006.

[111] A. Chau, S. Rignier, A. Delchambre, and P. Lambert. Three-dimensional model for capillary nanobridges and capillary forces. *Modelling and Simulation in Materials Science and Engineering*, 15(3):305–317, 2007.

Appendix A

A.1 SMPS probe dilution calibration

The SMPS probe sampler was calibrated by measuring the input flow vs. the flow that was sucked in through the capillary tube. The capillary tube flow (mL/min) was measured with a calibration standard (BIOS, DC-1 DryCal), while the dilution flow (L/min) which was let into the probe was measured with a bulk gas flow meter. Figure A.1 shows the calibration curves of the flow vs. the pressure difference at the reduction valve. All measurements were carried out at 298 K as it was not possible to measure the flow-rate through the sampler capillary at higher temperatures. To correct for the temperature another approach was made, which is presented in A.2. The dilution ratio (ξ) is calculated as:

$$\xi = \frac{Q_{sampler}}{Q_{probe}} \tag{A.1}$$

The calculated dilution ratio is seen on figure A.2.

A.2 Dilution ratio dependency on temperature

The temperature of the sampled gas affects the dilution ratio, since a higher temperature increases the viscosity. Therefore, a higher temperature decrease the flow-rate of the sampled gas, and increases the effective dilution ratio. The flow velocity in the capillary tube yields Reynolds numbers which are in the laminar range. The pressure-drop ΔP_{sc} across the capillary tube can be assumed independent on the temperature of the sampled aerosol, as the flow rate of the dilution gas does does not increase. The Hagen-Poiseuille equation can then be used to estimate the flow-rate in the sampler capillary tube, once the flow-rate at room temperature ($Q_{sampler}(T = T_0)$) is known:

$$Q_{sampler}(T = T_0) = \frac{\pi R_{sc}^4}{8\mu_g(T = T_0)} \frac{\Delta P_{sc}}{L_{sc}} \tag{A.2}$$

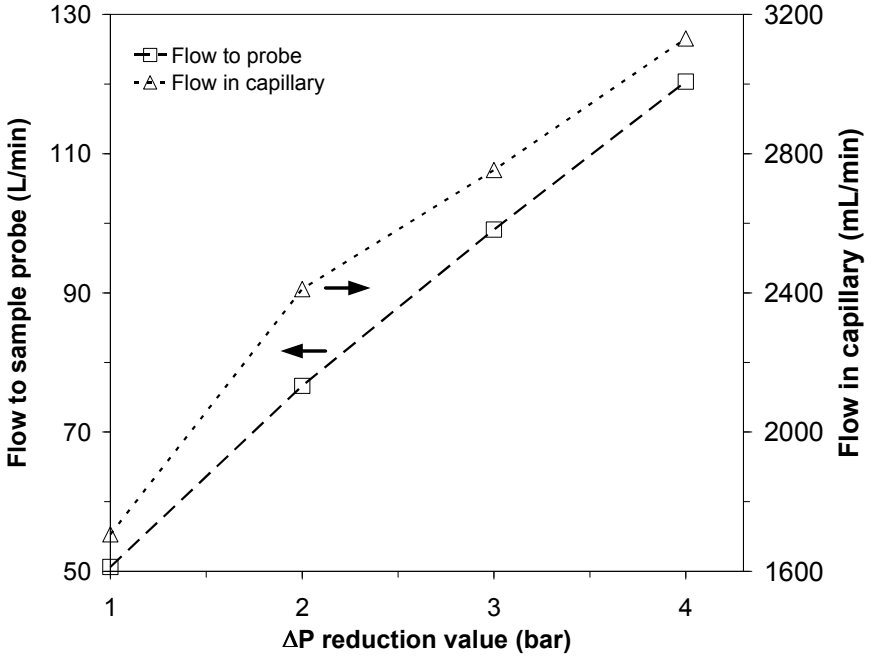


Figure A.1 Calibrated flow in probe and flow in capillary (sampled aerosol) at various pressure-drops in the reduction valve. As a standard for the SMPS measurements, ΔP was always chosen as 4 bar.

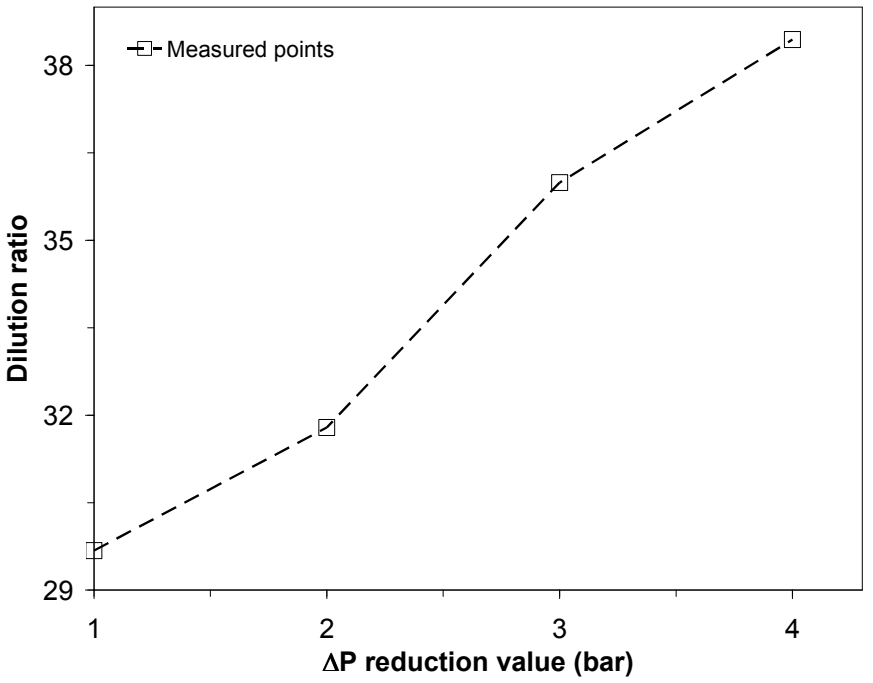


Figure A.2 The dilution ratio calculated from the measured values shown in figure A.1. An increase in the dilution ratio is seen for increasing ΔP.

where R_{sc} is the radius of the sampler capillary, $\mu_g(T = T_0)$ is the gas viscosity at room temperature (T_0), and L_{sc} is the length of the sampler capillary.

The ratio of the flow-rate at temperature T to the flow-rate at temperature T_0 is then:

$$\frac{Q_{sampler}(T)}{Q_{sampler}(T = T_0)} = \frac{\mu_g(T = T_0)}{\mu_g(T)} \tag{A.3}$$

The corrected dilution ratio then becomes:

$$\xi(T) = \xi(T = T_0)\frac{Q_{sampler}(T)}{Q_{sampler}(T = T_0)} = \xi(T = T_0)\frac{\mu_g(T = T_0)}{\mu_g(T)} \tag{A.4}$$

Appendix B

B.1 Precursor properties

The alumina precursor used was aluminum-acetylacetonate from Sigma Aldrich. The vapour pressure is [24, 23]:

$$lnP_s = 23.71 - \frac{13027}{T} \tag{B.1}$$

where P_s is the saturation pressure in *atmospheres* and T is the Kelvin temperature.

Table B.1 shows various properties for Al-acac taken from Sigma-Aldrich.

Table B.1 Properties of aluminum-acetylacetonate (Al_2O_3 precursor).

CAS-number -	Molar weight g/mol	Boiling point °C	Melting point °C
13963-57-0	324.31	315	190-193

Appendix C

C.1 Aerosol mass-concentration

The mass-concentration of the alumina aerosol may be determined by a mass-balance. Once the mass-concentration is known, it is possible to calculate the total number-concentration of the aerosol, assuming a size-distribution or monodisperse particles of a given size. The total number-concentration along with temperature profile in the flame reactor, is one of two major parameters which determines the structure of the end particles.

If the flow to the saturater is Q_c, the saturation pressure is P_s at saturater temperature T_s, and the total flow in the flame-reactor is $Q_t = Q_c + Q_{air} + Q_{quench} + Q_{\mathrm{CH_4}}$ (all at room temperature), then the total Al_2O_3 mass-concentration m_{tot} (kg/m^3) at room temperature T_0 becomes:

$$m_{tot} = \frac{M_{alumina}}{2} \frac{P_s(T_s)}{RT_0} \frac{Q_c}{Q_t} \tag{C.1}$$

where $M_{alumina}$ is the molar weight of alumina (101.96 g/mol). The factor $\frac{1}{2}$ comes since two Al atoms are used per molecule Al_2O_3.

At $T_s = 140°$C, the saturation pressure is (from appendix B.1:

$$P_s = e^{(23.71 - \frac{13027}{140+273.15})} \cdot 101325 = 40.7 \text{ Pa}$$

With the flame settings of table 2.4, the mass-concentration then becomes:

$$m_{tot} = \frac{0.10196}{2} \frac{40.7}{8.3145 \cdot 298.15} \frac{0.8}{20.44} = 3.27 \cdot 10^{-5} \text{ kg/m}^3$$

Appendix D

D.1 Characterization of *in-situ* annealed ceramic cakes

During the stay at the ETH Zürich, a novel method for cake-consolidation, termed *in-situ* annealing, developed by the PTL group was tested [11] on filtration made cakes. Flame annealing uses a particle-free spray flame to provide a rapid heating of the cake for times between 10 - 30 seconds. This highly non-isothermal sintering method has been seen to transform the morphology of SnO_2 cakes from the typical "lace-like" structure into what could best be described as a "cauliflower-like" structure, with significant increase in the mechanical stability as a result thereof [11].

Here the permeability method was applied in order to characterize the development of the morphology with the deposition-annealing cycles. Furthermore a simple model was set-up to determine the actual temperature through the cake during flame-annealing, as this is expected to be highly non-isothermal and dependent on the initial cake thickness.

D.2 Theory

D.2.1 Surface temperature during *in-situ* annealing

During *in-situ* annealing, which takes place as shown on figure D.1, the water cooled tube at which the substrate and cake is placed on, is continuously rotated. This is to avoid too large thermal stresses building up, which can crack the substrate along its length axis. The effect of rotation shall be ignored in the following analysis, as this should be fast enough in order to assume steady-state heat transfer. The flow of cooling water, as well as the thermal conductivity of the steel tube which holds the substrate, is large enough in order to maintain a constant temperature on the inside of the substrate of 25°C.

Figure D.2 shows a two layered structure (cake and substrate) through which heat conduction occurs. The boundary conditions are shown as well

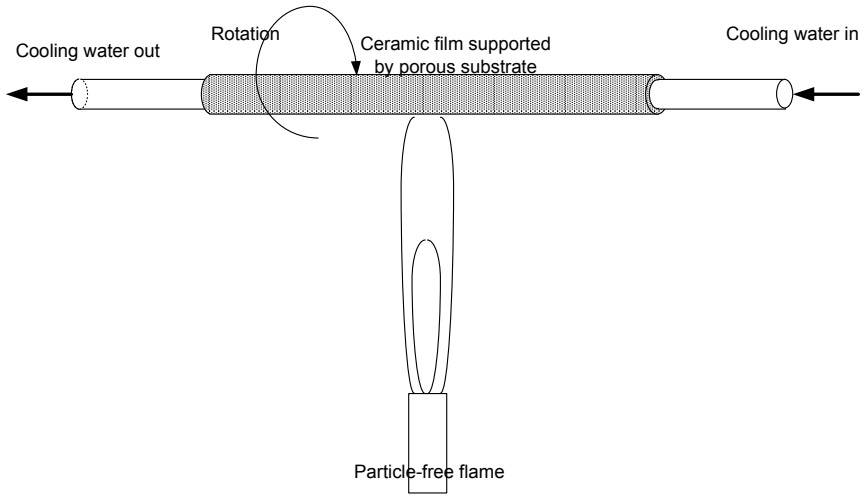


Figure D.1 Schematic of the flame annealing process. The substrate with the cake deposited on the outside, is placed on a water-cooled tube. A particle-free spray flame is a applied and the who setup is rotated to avoid thermal stresses building up.

as the coordinate system.

The temperature profile through the two layered structure is found by applying Fourier's law. The heat transported (Q) through the substrate (in cylindrical coordinates) becomes:

$$Q = 2\pi r L k_{eff,s} \frac{\mathrm{d}T}{\mathrm{d}r} \tag{D.1}$$

where r is the radial coordinate, and $k_{eff,s}$ is the effective thermal conductivity through the substrate (alumina), which can be described by [104]:

$$k_{eff,s} = k_{s,0} \left(1 + \frac{3\epsilon_s}{\left(\frac{k_g + 2k_{s,0}}{k_g - k_{s,0}}\right) - \epsilon_s} \right) \tag{D.2}$$

where $k_{s,0}$ is the bulk thermal conductivity of the solid material in the substrate (alumina), and k_g is the thermal conductivity of the gas contained within the porosity of the substrate. Equation D.2 is valid for low porosities. The gas thermal conductivity is typically very small and can be neglected, which simplifies eq. D.2 to:

$$k_{eff,s} \approx k_{s,0} \left(1 - \frac{3\epsilon_s}{k_{s,0} + \epsilon_s} \right) \tag{D.3}$$

The heat transported through the cake is:

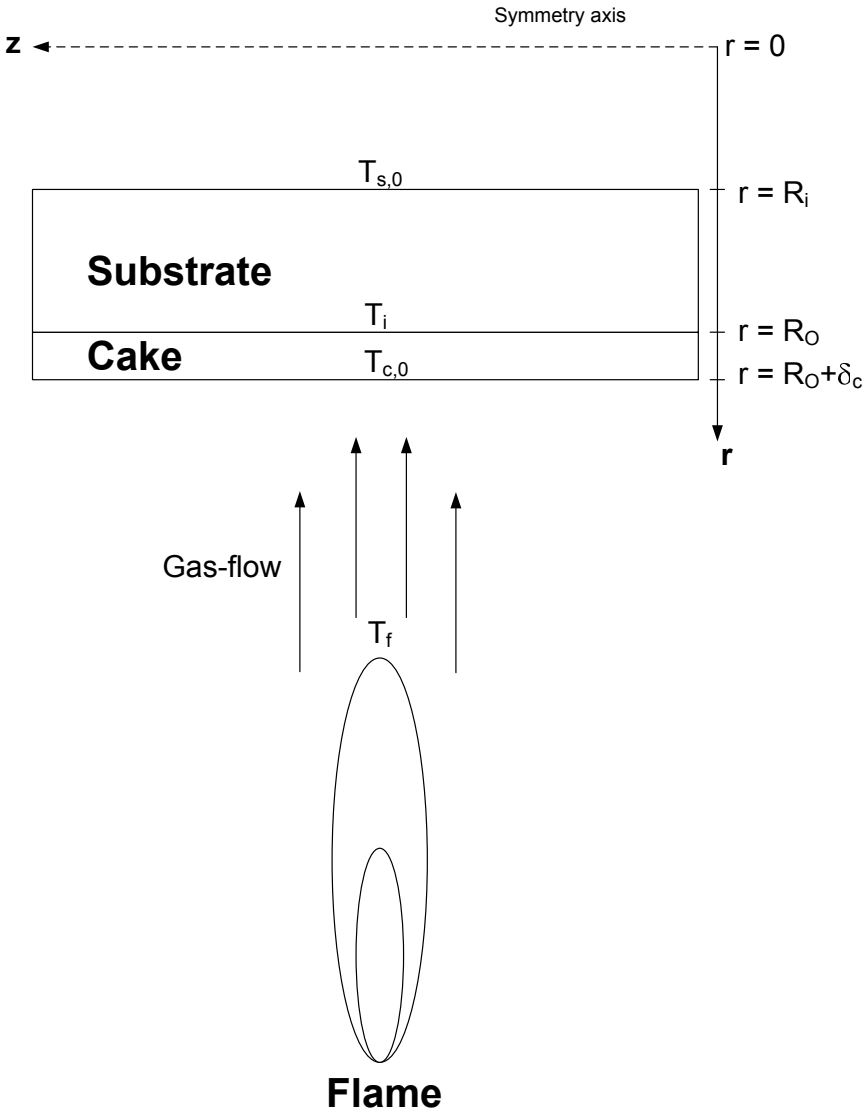


Figure D.2 Model domain with boundary conditions.

$$Q = 2\pi r L k_{eff,c} \frac{\mathrm{d}T}{\mathrm{d}r} \tag{D.4}$$

The thermal conductivity through the cake can not be estimated from eq. D.2, due to the high porosity of the cake. An equation describing the effective thermal conductivity through a highly porous medium is [105]:

$$\begin{aligned} k_{eff,c} =& \frac{1}{4}(k_g\left(3\epsilon_c - 1\right) + k_{c,0}\left(3(1-\epsilon_c) - 1\right) + \\ & \sqrt{\left[k_g\left(3\epsilon_c - 1\right) + k_{c,0}\left(3(1-\epsilon_c) - 1\right)\right]^2 + 8k_g k_{c,0}}\ ) \end{aligned} \tag{D.5}$$

where $k_{c,0}$ is the thermal conductivity of the solid phase of the cake (equal to $k_{s,0}$ since both materials are alumina.) Since ϵ_c is close to 1, the thermal conductivity of the gas can not be neglected. Unfortunately this is generally a function of the temperature, however in this analysis, it shall be assumed that it is a constant, determined as the average between the k_g at $T = T_{c,0}$ and k_g at $T = T_{s,0}$.

The boundary conditions for this coupled system of differential equations is:

$$\begin{aligned} &T(r = R_i) = T_{s,0} = 298\ \mathrm{K} \quad \text{Cooled inner tube} \\ &Q = \underbrace{h2\pi(R_O + \delta_c)\left(T_f - T_{c,0}\right)}_{\text{Convective heat transport from flame}} + \underbrace{2\pi(R_O + \delta_c)\varepsilon_c\sigma\left(T_f^4 - T_{c,0}^4\right)}_{\text{Radiation from flame}} \end{aligned} \tag{D.6}$$

where T_f is the temperature of the flame below the cake surface, ε_c is the emissivity of the alumina cake, σ is the Stefan-Boltzmann constant, and h is the heat transfer coefficient, which for a cylinder in a flow-field can be found through the Nusselt number as [106]:

$$Nu = \frac{2R_O h}{k_g} = \left(0.4Re^{\frac{1}{2}} + 0.06Re^{\frac{2}{3}}\right) Pr^{0.4} \left(\frac{\mu_{g,f}}{\mu_{g,0}}\right)^{\frac{1}{4}} \tag{D.7}$$

where Re is the Reynolds number, Pr is the Prandlt number, $\mu_{g,f}$ is the gas viscosity of the gas *below* the cake surface, and $\mu_{g,0}$ is the gas viscosity *at* the cake surface. The Reynolds number is:

$$Re = \frac{2R_O \rho_{g,\infty} U_g}{\mu_{g,\infty}} \tag{D.8}$$

where $\rho_{g,\infty}$ is the gas density below the cake surface, and U_g is the gas velocity past the cylinder. For turbulent free jets issuing from a round nozzle of diameter d_n, an expression for the (center stream-line) gas-velocity at a given distance (x) from the nozzle is [107]:

$$U_g = \begin{cases} U_0 & \frac{x}{d_n} \leq 5.27 \\ \frac{5.27 U_0}{\frac{x}{d_n}} & \frac{x}{d_n} > 5.27 \end{cases} \tag{D.9}$$

In the FSP, free jets typically develop above 2 cm from the spray nozzle (not to be confused with the jet "nozzle") [108]. The jet nozzle is therefore much larger than the spray nozzle. In the following, it shall be assumed that the jet nozzle diameter is 1 cm, an estimate based on the findings of Heine et al. [108].

The Prandtl number is:

$$Pr = \frac{C_{p,g,\infty}\mu_{g,\infty}}{k_{g,\infty}} \tag{D.10}$$

where $C_{p,g,\infty}$ and $k_{g,\infty}$ is the specific heat capacity and thermal conductivity for the gas below the cake surface, respectively.

The gas is assumed to be a mixture of CO_2 and H_2O, which are the main products from the combustion of methane (CH_4) and xylene (C_8H_{10}):

$$\begin{aligned} \mathrm{CH_4} + 2\mathrm{O_2} &\rightarrow \mathrm{CO_2} + 2\mathrm{H_2O} \\ \mathrm{C_8H_{10}} + \frac{21}{2}\mathrm{O_2} &\rightarrow 8\mathrm{CO_2} + 5\mathrm{H_2O} \end{aligned} \tag{D.11}$$

Properties for the gas is calculated using the following mixing rules:

$$X_g = y_{\mathrm{CO_2}} X_{\mathrm{CO_2}} + (1 - y_{\mathrm{CO_2}}) X_{\mathrm{H_2O}} \tag{D.12}$$

where X is the denotes the property, and $y_{\mathrm{CO_2}}$ is the molar fraction of CO_2 in the combustion gas which is (assuming no entrainment of surrounding gas):

$$y_{\mathrm{CO_2}} = \frac{F_{\mathrm{CH_4}} + 8F_{\mathrm{xylene}}}{3F_{\mathrm{CH_4}} + 13F_{\mathrm{xylene}}} \tag{D.13}$$

where F_i denotes the molar-flow (moles/s) of component i.

The temperature dependency of the properties of CO_2 and H_2O is calculated from the following interpolation formulas (CRC Handbook of Chemistry and Physics):

$$\begin{aligned} C_{p,\mathrm{CO_2}}(T/K) = \; & 24.99735 + 55.18696(T/1000) \\ & -33.69137(T/1000)^2 + 7.948387(T/1000)^3 \\ & -0.136638/((T/1000)^2) \\ & [\mathrm{J/molK}] \end{aligned} \tag{D.14}$$

$$\begin{aligned} \mu_{g,\mathrm{CO_2}}(T/K) = \; & -1.3276 \cdot 10^{-11} T^2 \\ & +5.4679 \cdot 10^{-8} T - 9.3916 \cdot 10^{-8} \\ & [\mathrm{kg/ms}] \end{aligned} \tag{D.15}$$

$$\begin{aligned} k_{g,\mathrm{CO_2}}(T/K) = \; & 1.3571 \cdot 10^{-8} T^2 \\ & +6.9843 \cdot 10^{-5} T - 5.06 \cdot 10^{-3} \\ & [\mathrm{W/mK}] \end{aligned} \tag{D.16}$$

$$\begin{aligned} C_{p,\mathrm{H_2O}}(T/K) = \; & 30.092 + 6.832514(T/1000) \\ & +6.793435(T/1000)^2 - 2.53448(T/1000)^3 \\ & +0.082139/((T/1000)^2) \\ & [\mathrm{J/molK}] \end{aligned} \tag{D.17}$$

$$\begin{aligned} \mu_{g,\mathrm{H_2O}}(T/K) = \; & 8.2327 \cdot 10^{-13} T^2 \\ & +3.8755 \cdot 10^{-8} T - 2.0429 \cdot 10^{-6} \\ & [\mathrm{kg/ms}] \end{aligned} \tag{D.18}$$

$$\begin{aligned} k_{g,\mathrm{H_2O}}(T/K) = \; & 7.5 \cdot 10^{-8} T^2 \\ & +2.63 \cdot 10^{-5} T + 4.19 \cdot 10^{-3} \\ & [\mathrm{W/mK}] \end{aligned} \tag{D.19}$$

Integration of eq. D.1 and D.4 yields expressions for the heat transported:

$$Q = \frac{2\pi L k_{eff,s} \left(T_i - T_{s,0}\right)}{\ln\left(\frac{R_O}{R_i}\right)} \tag{D.20}$$

$$Q = \frac{2\pi L k_{eff,c} \left(T_f - T_{c,0}\right)}{\ln\left(\frac{R_O+\delta_c}{R_i}\right)} \tag{D.21}$$

where T_i is the interface temperature between the two layers. Combining eq. D.20 and D.21, the unknown interface temperature may be found:

$$T_i = \frac{\frac{k_{eff,c}T_{c,0}}{\ln\left(\frac{R_O+\delta_c}{R_i}\right)} + \frac{k_{eff,s}T_{s,0}}{\ln\left(\frac{R_O}{R_i}\right)}}{\frac{k_{eff,c}}{\ln\left(\frac{R_O+\delta_c}{R_i}\right)} + \frac{k_{eff,s}}{\ln\left(\frac{R_O}{R_i}\right)}} \tag{D.22}$$

The surface temperature of the cake, $T_{c,0}$ is found from the boundary conditions (eq. D.6):

$$\begin{aligned} Q &= 2\pi L(R_O + \delta_c)\left(h(T_f - T_{c,0}) + \varepsilon_c\sigma(T_f^4 - T_{c,0}^4)\right) \\ &= \frac{2\pi L k_{eff,s}\left(T_i - T_{s,0}\right)}{\ln\left(\frac{R_O}{R_i}\right)} \end{aligned} \tag{D.23}$$

Equation D.23 cannot be solved analytically and has to be solved numerically. Inserting T_i into eq. D.23 yields:

$$\begin{aligned} Q &= (R_O + \delta_c)\left(h(T_f - T_{c,0}) + \varepsilon_c\sigma(T_f^4 - T_{c,0}^4)\right) \\ &= \frac{k_{eff,s}}{\ln\left(\frac{R_O}{R_i}\right)}\left(\frac{\frac{k_{eff,c}T_{c,0}}{\ln\left(\frac{R_O+\delta_c}{R_i}\right)} - \frac{k_{eff,c}T_{s,0}}{\ln\left(\frac{R_O+\delta_c}{R_i}\right)}}{\frac{k_{eff,c}}{\ln\left(\frac{R_O+\delta_c}{R_i}\right)} + \frac{k_{eff,s}}{\ln\left(\frac{R_O}{R_i}\right)}}\right) \end{aligned} \tag{D.24}$$

which can be arranged to yield:

$$(R_O + \delta_c)\left(h(T_f - T_{c,0}) + \varepsilon_c\sigma(T_f^4 - T_{c,0}^4)\right) = \frac{k_{eff,s}(T_{c,0} - T_{s,0})}{\ln\left(\frac{R_O+\delta_c}{R_i}\right) + \ln\left(\frac{R_O}{R_i}\right)} \tag{D.25}$$

Defining the error function E as:

$$E = (R_O + \delta_c)\left(h(T_f - T_{c,0}) + \varepsilon_c \sigma (T_f^4 - T_{c,0}^4)\right)\left(\ln\left(\frac{R_O + \delta_c}{R_i}\right) + \ln\left(\frac{R_O}{R_i}\right)\right) - k_{eff,s}(T_{c,0} - T_{s,0}) \quad \text{(D.26)}$$

the unknown surface temperature $T_{c,0}$ may be found by minimizing E.

The model parameters is shown in table D.1. Flows and densities are calculated assuming ideal gas properties.

Table D.1 Parameters used in the determination of the surface temperature during *in-situ* annealing.

Parameter	Value	Unit	Description
d_n	1	cm	Jet nozzle diameter
$x_{\text{annealing}}$	14	cm	Distance from nozzle to annealing position
U_0	330	m/s	Gas velocity at nozzle
T_f	1000	°C	Temperature below cake
$T_{s,0}$	25	°C	Temperature at inside
$k_{s,0}$, $k_{c,0}$	15.29	W/mK	Thermal conductivity of alumina
ϵ_c	0.95	-	Porosity of cake
ϵ_s	0.3	-	Porosity of substrate
Q_{CH_4}	1.23	L/min	Flow of methane (gas at 273.15 K)
Q_{xylene}	11.2	mL/min	Flow of xylene (liquid)

D.2.2 Experimental

Cakes were deposited by the filtration method, as described in chapter 3. The particles (TiO_2) were produced using a flame-spray pyrolysis (FPS) setup described elsewhere [11]. The expected mass-concentration and flame-settings can be seen in table D.2. Opposite the cakes of chapter 3, the cakes produced with the FSP setup was deposited on the *outside* of the ceramic substrate tubes (α-alumina, characterized in section 3.3.2).

After deposition and characterization by the permeability method, the *in-situ* annealing was carried out, by first placing the substrate containing the deposited cake on a water cooled tube. The outer diameter of this tube was 6 mm, which fitted well the internal diameter of the substrate tube. The cake was then placed over the FSP nozzle at a fixed distance (14 cm). A spray flame (particle free) of pure xylene was applied, while the water cooled tube and substrate was continuously turned around in that period. The height of the flame was approximately 13-14 cm (measured from nozzle). After 2 minutes, the xylene supply was cut and annealing was halted. The cakes were then characterized by the permeability method, and deposition - characterization - annealing - and characterization was repeated.

Table D.2 Flame settings for the FSP and flame-annealing experiments. The precursor (TTIP) was dissolved in xylene to a concentration of 0.5 M. Experiment settings are comparable to literature [11].

Setting	Value	Unit
CH_4 flamelet flow	1.5	L/min
O_2 flamelet flow	3.2	L/min
O_2 dispersion flow	5	L/min
O_2 sheath flow	5	L/min
Precursor flow	5	mL/min
Aerosol mass-concentration $(TiO_2)^{(1)}$	$1.17 \cdot 10^{-2}$	kg/m^3
Deposition time	20	s
Annealing time	2	min

$^{(1)}$: Calculated assuming no entrainment. Concentration at room temperature.

D.2.3 Results - characterization

Figure D.3 shows the cycle development of the cake capillary size. One cycle is defined as deposition with a subsequent annealing step.

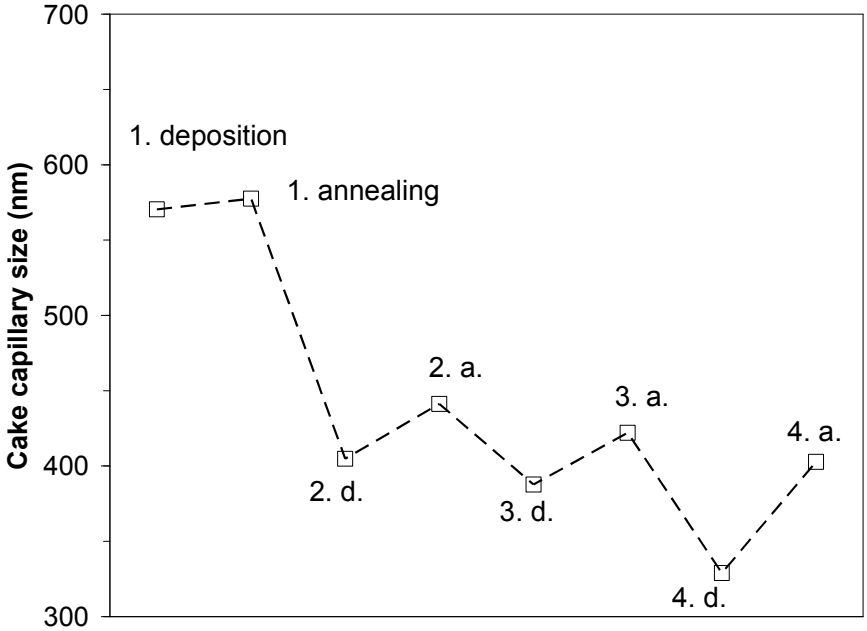


Figure D.3 Results of the permeability analysis for several deposition-annealing cycles.

The cake capillary size decreases with each deposition step and increases with each annealing step. The increase during annealing is most likely due to restructuring effects previously observed with this annealing method [11]. Unfortunately, SEM images are inconclusive, probably due to poor sample

handling which scratched or damaged the cakes.

It would appear, that an asymptotic behaviour is reached after 1 cycle. This is probably since particles do not penetrate the annealed cake, since they do also not penetrate the substrate capillaries, which are orders of magnitude larger than the cake capillaries. A new cake then builds up on top of the annealed cake with the morphology determined by the Pe number, which is approximately constant. Therefore, the morphology of the new cake is also constant, and an asymptotic behaviour is reached.

The initial cake thickness is not known, as it was not possible to measure any weight difference before and after deposition. This is possibly due to break-off of small amounts of the substrate when removing it from the deposition cell. Furthermore, the entrainment of gas is expected to be high [108], which weakens any estimate based on flow and mass-concentration. However, since after deposition, the cakes were observed as "bump" on the substrate, the thickness must have been in the order of 100 μm and upwards.

The cake average capillary size is larger (approximately 2-3 times) than that of cakes formed with lower aerosol concentrations (see chapter 3). Due to the high mass-concentration (approximately 1000 times that of the premixed-flame), large agglomerates are expected to be present in the aerosol. The deposition of agglomerates have been shown to increase the final porosity of cakes in models of flat substrate deposition [10]. Even though the geometry is different, chapter 5 showed, that the same porosity as that of flat substrate deposited cakes was also achieved for porous substrates when cake filtration began. Therefore, the cake capillary size is also expected to be higher when large agglomerates are present.

D.2.4 *In-situ* annealing temperature

Figure D.4 shows the calculated surface temperature of the cake during *in-situ* annealing. Model parameters are given in D.2.

The surface temperature varies greatly with the thickness of the cake, due to the low thermal conductivity of the highly porous cake. This is important to keep in mind, whenever *in-situ* annealing is used, since cakes of different thickness will experience very different temperatures. This makes it difficult to fix any process parameter for later analysis. Since the thickness of the cakes were expected to be > 100 μm, the surface temperature most likely reached well over 700°C during *in-situ* annealing.

The analysis is only valid for non-shrinking cakes, where the porosity remains constant. This is true initially, however shortly after the onset of *in-situ* annealing, these two parameters will change. The above model should therefore only be used to obtain the initial surface temperature.

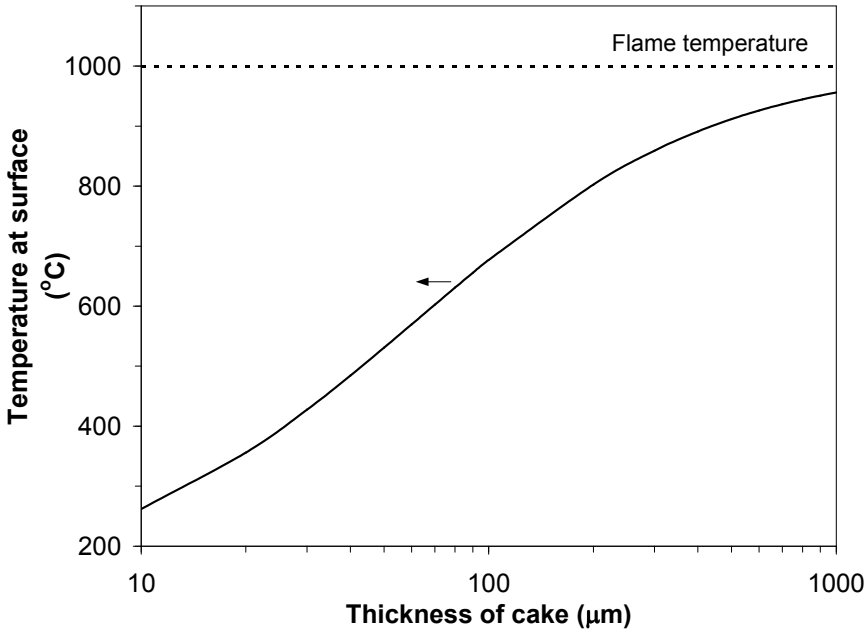


Figure D.4 Calculated surface temperature during *in-situ* annealing as a function of the initial cake thickness.

D.2.5 Conclusions

Porous cakes of FSP-made TiO_2 were deposited on the outside of porous α-alumina substrates by the filtration method. *In-situ* annealing was applied on the surface of the as-deposited cakes and characterization of the deposition-annealing cycles using the permeability method was carried out. The high concentration of TiO_2 particles increased the average capillary size of the cake to more than what was expected from studies at lower aerosol concentrations, possibly due to large agglomerates in the filtration gas.

The cake average capillary size grew after annealing, most likely due to restructuring effects. Repeating the deposition lead to a decrease in the cake capillary size, due to the immediate formation of a cake with smaller permeability. After a few cycles of deposition and annealing steps, the cake average capillary size decreased toward an asymptotic value.

The surface temperature of the cake during *in-situ* annealing was found to increase greatly with increasing cake thickness. The expected initial surface temperatures was above 700°C.

Appendix E

E.1 Ratio of adhesive forces

The Van der Waals force between two identical (spherical) particles of diameter d_p is [109]:

$$F_{vdw} = \frac{A d_p}{12 a^2} \tag{E.1}$$

where A is the Hamakar constant equal to $15 \cdot 10^{-20}$ J for alumina [110], and a is the contact distance typically set to 0.3 - 0.4 nm. The compressive capillary forces acting when a liquid meniscus forms between two identical spherical particles (diameter d_p) touching each other is [85]:

$$F_{cap} = \sigma \left(\frac{\pi}{4} d_p^2 \sin^2 \phi \left(\frac{1}{\rho_2} - \frac{1}{\rho_1} \right) + \pi d_p \sin \phi \cdot \sin(\phi + \theta) \right) \tag{E.2}$$

where σ is the surface tension, ϕ is the angle from the center of the particles to the wetted perimeter of the particles, and θ is the contact angle, which is often assumed to be 0. The principle radii, ρ_1 and ρ_2 is given as:

$$\rho_1 = \frac{d_p}{2} \sin \phi \frac{d_p}{2} \left(1 - \cos \phi\right) \frac{1 - \sin(\phi + \theta)}{\cos(\phi + \theta)} \tag{E.3}$$

and

$$\rho_2 = \frac{d_p}{2} \frac{1 - \cos \phi}{\cos(\phi + \theta)} \tag{E.4}$$

If the contact angle is assumed zero, the angle ϕ may be determined from the Kelvin equation since the sum of the inverse radii of curvature is equal to the inverse Kelvin radius (r_k) [111]. In eq. E.2, the 1st principle radius of curvature was defined negatively, therefore r_k is equal to:

$$\frac{1}{r_k} = \left(\frac{1}{\rho_2} - \frac{1}{\rho_1}\right) = -\frac{RT \ln S}{2\sigma V_m} \tag{E.5}$$

where V_m is the molar volume of the condensing liquid. Inserting eq. E.3 and E.4 into E.5 (with $\theta = 0$) yields:

$$\frac{2 \sin\phi \cos\phi}{d_p(1 - \cos\phi)(\cos\phi + \sin\phi - 1)} = -\frac{RT \ln S}{2\sigma V_m} \tag{E.6}$$

which may be solved for ϕ once S is known.

Appendix F

F.1 Filtration theory

F.1.1 Cake filtration at constant flow

When the cake-filtration sets in, a particle layer with constant solid volume fraction builds up outside of the capillary plug as indicated on F.1.

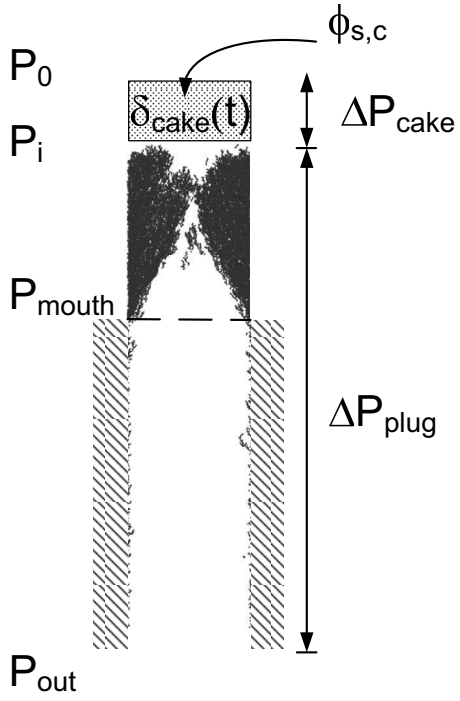


Figure F.1 : Constant solid volume fraction filtration on top of a plugged capillary. The pressure at the inlet to the filter is $P_0 = 1$ bar and the pressure at the outlet (P_{out}) is determined by the structure of the deposis formed during filtration.

Pressure-drop through the cake (and outside of the capillary in general) is modeled using d'Arcy's law. For the cake with constant solid volume

fraction ($\phi_{s,c}$) this becomes:

$$P\frac{dP}{dz} = -\frac{Q_0 P_0 \mu_g}{AB_{0,c}} \tag{F.1}$$

where Q_0 is the flow at pressure $P_0 = 1$ bar, μ_g is the gas viscosity, A is the filtration area equal to πR_{c0}^2 and $B_{0,c}$ is the d'Arcy permeability at constant solid volume fraction. Integration from P_0 (inlet pressure) to P_i(interface pressure) through the cake yields:

$$P_i^2 - P_0^2 = -\frac{2Q_0 P_0}{AB_{0,c}}\delta(t) \tag{F.2}$$

where δ(t) is the thickness of the cake at time t. Since the capillary is plugged, all particles deposit within on top of the growing cake. A mass-balance then gives the thickness of the cake with time:

$$\delta(t)\phi_{s,c}A = QC_n v_p t \tag{F.3}$$

Inserting (F.3) in (F.2) yields:

$$\Delta P_{cake} = P_0 - P_i = P_0 - \sqrt{P_0^2 - \frac{2Q_0^2 P_0 \mu_g}{A^2 B_{0,c}\phi_{s,c}} C_n v_p t} \tag{F.4}$$

where the cake constant solid volume fraction ($\phi_{s,c}$) is found from the LD model at the corresponding Pe value.

The total pressure drop can be found by adding the pressure-drop of the plug to the pressure-drop of the cake. D'Arcy's law for the plug outside of the capillary is:

$$\frac{1}{2}\left(P_i^2 - P_{mouth}^2\right) = \frac{Q_0 P_0 \mu_g}{A} \int_{z_{mouth}}^{z_i} \frac{dz}{B_0(z)} = \mathrm{k}_1 \tag{F.5}$$

The RHS of (F.5) is constant (k_1) in time, as the particles do not change the morphology of the deposit below the cake after plugging. The pressure-drop through the capillary is:

$$\frac{1}{2}\left(P_{mouth}^2 - P_{out}^2\right) = \frac{8Q_0 \mu_g P_0}{\pi R_{c,0}^4} \int_{z_{mouth}}^{z_i} \frac{dz}{(1-\phi_s(z))^2} = \mathrm{k}_2 \tag{F.6}$$

where k_2 is also constant in time. Adding (F.5) and (F.6) yields the pressure-drop through the total plug:

$$\frac{1}{2}\left(P_i^2 - P_{out}^2\right) = \mathrm{k}_1 + \mathrm{k}_2 \tag{F.7}$$

At the time of capillary plugging, $P_i = P_0$ and the total pressure-drop across the plug is known ($\Delta P_{plug}(t_{plug}) = P_0\text{-}P_{out}(t_{plug})$). Therefore, one can find k_1+k_2 by solving (F.7) at t_{plug}:

$$2(k_1 + k_2) = P_0^2 - P_{out}^2(t_{plug}) = P_0^2 - (P_0 - \Delta P_{plug}(t_{plug}))^2 \quad \text{(F.8)}$$

The pressure-drop through the plug is then:

$$\begin{aligned}\Delta P_{plug}(t) &= \underbrace{P_i}_{P_0-\Delta P_{cake}} - \overbrace{P_{out}}^{\sqrt{P_i^2-2(k_1+k_2)}} \\ &= P_0 - \Delta P_{cake} - \sqrt{(P_0 - \Delta P_{cake})^2 - 2(k_1 + k_2)}\end{aligned} \quad \text{(F.9)}$$

Therefore, the total pressure-drop can be found as:

$$\Delta P = \Delta P_{plug} + \Delta P_{cake} = P_0 - \sqrt{(P_0 - \Delta P_{cake})^2 - 2(k_1 + k_2)} \quad \text{(F.10)}$$

F.2 Constant pressure-drop filtration

If the pressure-drop is kept constant during filtration, the flow-rate at the filter surface (Q_0) will change with time. The pressure-drop through the particle layer deposited inside and outside of the capillary is once again modeled using d'Arcy's law and Poiseulle's law respectively. The equations are:

$$\frac{1}{2}(P_0^2 - P_{mouth}^2) = \frac{Q_0(t)\mu_g P_0}{\pi R_{c,0}^2} \underbrace{\int_{outside} \frac{dz}{B_0(t,z)}}_{I_1(t)} \quad \text{(F.11)}$$

$$\frac{1}{2}\left(P_{mouth}^2 - P_{out}^2\right) = \frac{8Q_0(t)\mu_g P_0}{\pi R_{c,0}^4} \underbrace{\int_{inside} \frac{dz}{\varepsilon^2(t,z)}}_{I_2(t)} \quad \text{(F.12)}$$

where the integrals on the RHS of eq. (F.11) and (F.12) have been shortened to $I_1(t)$ and $I_2(t)$ respectively. Both values can be found using the LD model and integrating using an appropriate numerical scheme. Summing eq. (F.11) and (F.12) yields the pressure-drop through both inside and outside:

$$\frac{1}{2}(P_0^2 - P_{out}^2) = \frac{Q_0(t)\mu_g P_0}{\pi R_{c,0}^2} I_1(t) + \frac{8Q_0(t)\mu_g P_0}{\pi R_{c,0}^4} I_2(t) \quad \text{(F.13)}$$

or expressed in terms of $Q_0(t)$:

$$Q_0(t) = \frac{\left(P_0^2 - P_{out}^2\right)\pi R_{c,0}^2}{2\mu_g P_0 \left(I_1(t) + \frac{8}{R_{c,0}^2} I_2(t)\right)} \tag{F.14}$$

As the total pressure-drop ($\Delta P = P_0 - P_{out}$) is known, eq, (F.14) is better expressed in terms of this as:

$$Q_0(t) = \frac{\left(2P_0\Delta P - \Delta P^2\right)\pi R_{c,0}^2}{2\mu_g P_0 \left(I_1(t) + \frac{8}{R_{c,0}^2} I_2(t)\right)} \tag{F.15}$$

As most particles are deposited outside of the capillary even at high Pe numbers, the pressure-drop increase due to capillary deposition may be neglected and $I_2(t)$ becomes constant I_2. The value of the constant I_2 is:

$$I_2 = \frac{\left(2P_0\Delta P - \Delta P^2\right)\pi R_{c,0}^4}{16 Q_0(t_0)\mu_g P_0} \tag{F.16}$$

where $Q_0(t_0)$ is the flowrate at the filter surface at time t_0 (initial flow-rate). Once the flowrate decreases due to the deposition, the flux of particles to the filter surfaces decreases as well. The time-step Δt between each particle that deposits then also changes.

F.3 Deposition model code

F.3.1 Main.f90

```
program capillary_clogging
use variables_deposition
    implicit none
    integer*4 coll,grid_0(3),grid_end(3),grid_coll(3),grid_temp(3)
    integer*4 cell_0,cell_end,i,l,empty_flag,simple_sign,overlapping,trash,position
    INTEGER, DIMENSION(:), ALLOCATABLE :: seedOld
    character*12 dummy1,startTime
!    integer :: n

    real*8 cell_number,test,grid_size
    real*8 rcstore(3),dold,d
    real*8 parti,partj
    real*8 r1(3),r2(3)
    real*8 rtest

! ----- Initialize variables etc.
    write(*,*) 'Initializing variables'
    call init_variables()
    open (unit=5,file='inserted.dat')
    open (unit=10,file='deposited_particles-time.dat')
    open (unit=15,file='deposited_particles-srt-by-cell.dat')
    open (unit=20,file='nl-error.dat')
    open (unit=25,file='specifications.dat')
    open (unit=30,file='dist-error.dat')
    open (unit=35,file='summery.dat')
    open (unit=40,file='penetration.dat')
    open (unit=45,file='timeused.dat')
    open (unit=50,file='temp.dat')
    open (unit=55,file='diffusivity.dat')

    call date_and_time(creation_date,creation_time)
    call check_continuation()
    call write_header_to_file()
! ----- Check if program should continue from previous finished calculations
    if (continuation.eq.1) then
      write(*,*) 'Continuing from previous calculation'
      call read_from_file()
      if (moving.eq.1) then
        goto 20
      else
        goto 10
      endif
    else
      original_creation_date = creation_date
      original_creation_time = creation_time
      write(*,*) 'Program started at: ', creation_time
      call init_random_seed()
  close(50)
      open(unit=50,file='temp.dat')
    endif

!  ----- Start main program

10  call drop_particle()
iter = 0
    call acc_dist(1)
    re(1) = r0(1)
    re(2) = r0(2)
    re(3) = r0(3)
!    stop_program = 1
    if (stop_program.eq.1) then
      goto 30
    endif

! ----- Move the particle after insertion. If the position after movement is negative in the z-direction,
!   then particle has moved out of bounds

15  call langevin_integration() ! calculate first step
    if (re(3).lt.0) then ! reintroduce particle at top if the first step takes it outside
      velocity(3) = -velocity(3)
      re = r0
      iter = iter+1
      goto 15
    endif
    if (filmdeposition.eq.1) then
      call wall_collision_check(re,coll)
      if (coll.eq.1) then
        velocity(1) = -velocity(1)
```

```
        velocity(2) = -velocity(2)
        lost = lost+1
        re = r0
        goto 15
      endif
    else
      call boundary_check(re,coll)
      if (coll.eq.1) then
        velocity(1) = -velocity(1)
        velocity(2) = -velocity(2)
        lost = lost+1
        re = r0
        goto 15
      endif
    endif


! ----- After moving the particle, the neighbour list is updated and collision is checked.
!   correct collision coordinates are also calculated

20  l = 1 ! indicator for particle collision. If l=1 collision between the wall is checked
    d = 1 ! size indicator for the distance travelled of the depositing particle to the possible collision partner
    dold = d

! Check for stop
    call check_time(position)
    if (position.eq.30) then
      goto 30
    endif
! *****
    call date_and_time(dummy1,startTime)
    call check_grid(re,grid_end)
    call check_nl(re,grid_end,empty_flag)

    if (empty_flag.ne.1) then
      do i = 1,size(neighbour_list,1)
        call collision_check(empty_flag,neighbour_list(i,:),coll)
        if (coll.eq.1) then
          call collision_coordinates(neighbour_list(i,:),rc,d)
          l = l+1
        endif
        if (d.lt.dold) then
          rcstore = rc
          dold = d
        endif
      enddo
endif
    call count_time(startTime,0)
    if (l.gt.1) then ! l>1 in case of any collisions
      rc = rcstore
      call check_grid(rc,grid_coll)
      call boundary_check(rc,coll) ! check if there is any collision with the borders
      if ((coll.ne.1).and.(filmdeposition.ne.1)) then
        if ((grid_coll(3).gt.skipIncludingCell).or.(skipfirst.ne.1)) then
          call wall_collision_check(rc,coll) ! possibility of simultaneous particle-particle-wall collision
          call wall_collision_coordinates(coll,rc) ! if so, then calculate new collision coordinates
        endif
      endif
      if ((coll.eq.1).and.(filmdeposition.eq.1)) then ! if all geometry is turned off check if particle is colliding with bottom
        call wall_collision_coordinates(coll,rc)
      endif

      if (nint(part_pos(1,1)).eq.0) then ! first particle deposits
        call store_first_particle()
        overlapping = 0
        call write_line_to_file()
        call number_balance()
        if (balance.ne.0) then
          write(*,*) 'Mass balance NOT valid: first particle deposit (p-p)'
          call number_balance()
          write(*,*) lost_bottom, lost_side,lost
        endif
      else
        call store_particle() ! stores particle in large matrix sorted by cell number
        call check_grid(rc,grid_coll)
! ----- for debugging
        call debug_collisions(1)
! ----- for debugging end

        call write_line_to_file()

! ---- for debugging
        call number_balance()
        if (balance.ne.0) then
```

```
          write(*,*) 'Mass balance NOT valid - p-p'
          rtest = dsqrt(rc(1)**2+rc(2)**2)
          write(*,*) rtest
!          read(*,*)
          call number_balance()
          write(*,*) lost_bottom, lost_side,lost
        endif
        call check_time(trash)
! ----- for debugging END

      endif
else ! if no collision between particles occur, check for collisions between particle and wall
      call wall_collision_check(re,coll)

! ----- excluding walls
      if (filmdeposition.eq.1) then
        if (coll.eq.1) then
          call boundary_check(re,coll) ! check collision with bottom
          if (coll.eq.1) then
            call wall_collision_coordinates(coll,rc)
          else
            re = r0
            velocity(1) = -velocity(1)
            velocity(2) = -velocity(2)
            lost=lost+1
            coll = 0
          endif
        else
          call boundary_check(re,coll)
          if (coll.eq.1) then
            call wall_collision_coordinates(coll,rc)
          endif
        endif
      else
! ----- including walls
        call boundary_check(re,coll)
        if (coll.ne.1) then
          if ((grid_end(3).gt.skipIncludingCell).or.(geometry.eq.0)) then
            call wall_collision_check(re,coll)
            call wall_collision_coordinates(coll,rc)
          endif
        else
          re = r0
          velocity(1) = -velocity(1)
          velocity(2) = -velocity(2)
          lost=lost+1
          coll = 0
        endif
      endif

      if (coll.eq.1) then
        if (nint(part_pos(1,1)).eq.0) then
          call store_first_particle()
          call acc_dist(0) ! calculate diffusivity
          overlapping = 0
          call write_line_to_file()
          call number_balance()
          if (balance.ne.0) then
            write(*,*) 'Mass balance NOT valid first particle p-w'
            call number_balance()
            write(*,*) lost_bottom, lost_side,lost
          endif
        else
          call store_particle()
          call acc_dist(0)
          call check_grid(rc,grid_coll)
! ----- For debugging
          call debug_collisions(1)
! ----- For debugging END

          call write_line_to_file() !  "
          call number_balance()
          call check_time(trash)
          if (balance.ne.0) then
            write(*,*) 'Mass balance NOT valid p-w'
            call number_balance()
            write(*,*) lost_bottom, lost_side,lost
          endif

        endif
      else
!  ----- In case of no collision with either wall or other particle the particle is moved by integration

50      r0 = re ! update position of particle
```

```
        call date_and_time(dummy1,startTime)
        call langevin_integration()

!  calculate diffusion coefficient - number of moves can be set by changing test_moves. Better statistics can be obtained by
! increasing the number of particles inserted by changing inserted_max.
        if (difcoef.eq.1) then
          if ((difcoef.eq.1).and.((test_moves.gt.10000).and.(inserted.le.inserted_max))) then ! Diffusion coefficient calc.
            call acc_dist(0)
            goto 10 ! -"-
          else
            if (inserted.gt.inserted_max) then
              goto 40 ! goto end
            else
              goto 50
            endif
          endif
        endif
! end calculation of diff. coeff.
        call count_time(startTime,1)
! ----- The resulting position in "z" is checked to see if the particle has moved out of bounds

        if (re(3).lt.0) then
          re = r0
          re(3) = 0d0
          velocity = 0d0
          goto 50
        endif

        if ((re(3).gt.zmax).and.(filmdeposition.ne.1)) then ! particle has moved out of the bounds- insert a new particle
          lost_bottom = lost_bottom+1
          call check_time(position)

          if ((nint(part_pos(1,1)).eq.0).or.(position.eq.10)) then ! to avoid check_time()
                                                                   ! from returning with value=20
                                                                   ! (which it will before first particle has deposited)
            goto 10
          endif
          if (position.eq.30) then
            goto 30
          endif

        endif

! ----- After moving the particle return in the loop to check once again for collision etc.

        call check_time(position)
        if ((nint(part_pos(1,1)).eq.0).or.(position.eq.20)) then
          goto 20 ! particle is moving and no stop is given
        endif
        if (position.eq.30) then
          goto 30 ! stop the program
        endif

      endif
    endif

! ----- A particle has been succesfully deposited, now check if the below criteria are fulfilled and continue if so

30  if ((dropped.lt.dropped_max).and.(stop_program.ne.1)) then ! continue as long as these criteria are fullfilled.
      if ((geometry.eq.0).and.(inserted.gt.inserted_max)) then
        stop_program = 1
      endif
      goto 10
    else
      call write_to_file()
      call write_to_temp()

    endif

!  ----End of program. Write a nice little notice indicating succesfull completion

    if (timeLimitReached.eq.1) then
      write(*,*) 'Program ended due to time limit reached'
    else
      write(*,*) 'Program ended succesfully'
      call write_summery()
      call write_penetration_list()
      call write_specifications()
      write(*,*) 'Files written'
    endif


40 end
```

F.3.2 Drop_particle.f90

```
subroutine drop_particle()
use variables_deposition
    implicit none
    integer*4 coll,i,empty_flag,grid_0(3),position,dummy
real*8 r,phi,x,y

10  call check_time(position)
    if (position.eq.30) then
      return
    endif
    call random_number(r)
    call random_number(phi)

    r = radius*dsqrt(r)
    phi = 2d0*pi*phi

    x = r*cos(phi)
    y = r*sin(phi)
    test_moves = 0
    if (difcoef.eq.1) then
      r0(1) = 1d-10
      r0(2) = 1d-10
      r0(3) = 0.5d0*zmax

    else
      r0(1) = x
      r0(2) = y
      r0(3) = 0d0
    endif

    rinlet = r0


    call check_grid(r0,grid_0) ! locate particle at initial position
    ncellold = 0
    call check_nl(r0,grid_0,empty_flag) ! check neighbour list at initial position. The flag empty_flag returns 1 if the list is empty
    call initial_boundary_check(coll)
    if (coll.eq.1) then
      goto 10
    endif

    if (empty_flag.ne.1) then
      do i =1,size(neighbour_list,1)
        call initial_collision_check(neighbour_list(i,:),coll)
        if (coll.eq.1) then ! if there is ANY overlap between the newly dropped particle and the old one, put in a new and count one more
          dropped = dropped+1
          inserted = inserted+1
          call check_time(dummy)
          if (stop_program.eq.1) then
            return
          endif
          goto 10
        endif
      enddo
    endif
    inserted = inserted+1
!    write(*,*) inserted
! Initialize velocity vector again
    velocity(1) = 0d0
    velocity(2) = 0d0
    velocity(3) = 0d0

end
```

F.3.3 Move_particle.f90

```
subroutine move_particle()
    use variables_deposition
    implicit none
    real*8 phi,theta,x,y,z,ugas,r

    actual_time = actual_time+dt


!    phi = random@()
!    theta = random@()
    call random_number(phi)
    call random_number(theta)

    r = sqrt((re(1)**2+re(2)**2))
```

```
    ugas = ugasmax*(1-(r/radius)**2)
!    ugas = ugasmax/2

    phi = 2*pi*phi
    theta = pi/2*theta
    theta = 0

    if (geometry.eq.0) then
      x = disp*d_0pseudo*cos(phi)
      y = disp*d_0pseudo*sin(phi)
      z=0
    else
      x = disp*d_0*cos(phi) !*cos(theta)
      y = disp*d_0*sin(phi) !*cos(theta)
!     z = 0.02*d_0*sin(theta)
      z= 0
    endif


    re(1) = re(1)+x
    re(2) = re(2)+y
    re(3) = re(3)+z+dt*ugas

end



! ***** Based on the solution method by Ermak & Buckholz, J. Comp. Phys., 35, 169-182, 1980
! ***** Adapted from M. Heine & S.E. Pratinis, Langmuir, 23, 9882-9890, 2007
! ***** Modified using the equations by Gutsch, Pratsinis and Löffler, J. Aerosol Sci., 26, 2, 187-189, 1995

subroutine langevin_integration()
use variables_deposition
    implicit none
    real*8 alpha,beta,G,H,I,y(6),rqh(3),vqh(3),ugas,r
    real*8 rnew(3),w(3)
    integer*4 j

actual_time = actual_time+dt
    r = dsqrt(re(1)**2+re(2)**2)
    w(1) = 0d0
    w(2) = 0d0
    w(3) = ugasmax/2d0

!    w(3) = ugasmax*(1d0-(r/radius)**2d0)



    if (geometry.eq.0) then
      alpha = 18d0*muFluid/(rhoPart*d_0pseudo**2d0*Cc) ! correct friction for slip
      beta = kB*Tgas/(pi/6d0*(d_0pseudo)**3d0*rhoPart)
    else
      alpha = 18d0*muFluid/(rhoPart*d_0**2d0*Cc)
      beta = kB*Tgas/(pi/6d0*(d_0)**3d0*rhoPart)
    endif

    G = beta*(1d0-dexp(-2d0*alpha*dt))
    H = beta*(1d0/alpha*(1d0-dexp(-alpha*dt))**2)
    I = beta*(1d0/(alpha**2d0))*(2d0*alpha*dt-3d0+4d0*dexp(-alpha*dt)-dexp(-2d0*alpha*dt))
    call gauss(y)
    if (geometry.eq.0) then ! no axial dispersion for geometry = 0

      do j = 1,3
        vqh(j) = dsqrt(G)*y(j)
        rqh(j) = H/dsqrt(G)*y(j) + dsqrt(I-H**2/G)*y(j+3)

        re(j) = rqh(j)+re(j)+velocity(j)/alpha*(1d0-dexp(-alpha*dt))+w(j)*(dt-1d0/alpha*(1d0-dexp(-alpha*dt))) ! update position
        velocity(j) = vqh(j)+velocity(j)*dexp(-alpha*dt)+w(j)*(1d0-dexp(-alpha*dt)) ! update velocity
      enddo

    else
      do j =1,3
        vqh(j) = dsqrt(G)*y(j)
        rqh(j) = H/dsqrt(G)*y(j) + dsqrt(I-H**2/G)*y(j+3)

        re(j) = rqh(j)+re(j)+velocity(j)/alpha*(1d0-dexp(-alpha*dt))+w(j)*(dt-1d0/alpha*(1d0-dexp(-alpha*dt))) ! update position
        velocity(j) = vqh(j)+velocity(j)*dexp(-alpha*dt)+w(j)*(1d0-dexp(-alpha*dt)) ! update velocity
      enddo
    endif
    test_moves = test_moves+1
    call diffusivity()
! ----- Parabolic flow profile used for validation to the Friedlander / Hinds
!         deposition model (Friedlander: "Smoke, Dust, and Haze", 2nd edition, 2000, page 79
```

```
!   and Hinds: "Aerosol Technology", 2nd edition, 1999, page 163.)


!   re(3) = re(3) + dt*ugas ! add fluid velocity component in the z-direction
end
```

F.3.4 Calculate_neighbourandgrid.f90

```
! This subroutine carries out all calculations required to check the neighbour list.
! It is called from the main program with inputs "P" containing the particle position and "GRID_POS" containing the position
! of the grid (in NR,NPHI,NZ).
! It calculates if the particle is in the vicinity of other grids after it has been moved. If so, then the grid number is noted down
! in order to build the neighbour list.
! It outputs a neighbour list from subroutine build_nl, that is called at the very end of this routine.

subroutine check_nl(p,grid_pos,empty_list)
use variables_deposition
    implicit none
    real*8 p(3),pCyl(3)
    real*8 xj,yj,zj,rgridj,cell_number
    real*8 z_grid0,z_grid1
    real*8 r_grid0,r_grid1
    real*8 phi_grid0,phi_grid1
real*8 z,r,phi
    real*8 dist
    integer*4 nz,nphi,nr,nrj,grid_pos(3),firstcellflag,numberCells,particles
    integer*4 empty_list,iStart,nlSize,nzj,nzk,nphil,grid_end(3),ncell
    integer*4 i,j,k,l,m ! counters
    integer*4, dimension(:), allocatable :: ipoint(:)
    integer*4, dimension(:), allocatable :: cell_list(:)
    integer*4 ncell0,grid_0(3)
    real*8 tol

    if (geometry.eq.0) then
      empty_list = 1
      return
    endif
    deallocate(debug_cell)

    m = 1
    tol = 1d-9
    nr = grid_pos(1)
    nphi = grid_pos(2)
    nz = grid_pos(3)

    call check_grid(r0,grid_0)
    call check_grid(p,grid_end)

    ncell0 = nint(cell_number(grid_0(1),grid_0(2),grid_0(3)))
    ncell = nint(cell_number(grid_end(1),grid_end(2),grid_end(3)))

    if (ncell.eq.ncellold) then ! Only generate new list if particle has moved from one cell to another (to save time)
      allocate(debug_cell(1,1))
      empty_list = empty_old
      return
    endif

    r_grid1 = nr*dr
    r_grid0 = (nr-1)*dr

    phi_grid1 = (nphi-1)*dphi ! since nphi is counted differently (at dphi>=phi>=0, nphi = 1)
    phi_grid0 = nphi*dphi

    z_grid1 = nz*dz
    z_grid0 = (nz-1)*dz
    call cylindrical_coordinates(p,pCyl)
r = sqrt(p(1)**2+p(2)**2)
    phi = pCyl(2)
    z = p(3)

    if (nr.le.nrcrit) then
      allocate(cell_list(3*nphimax*(nrcrit+1)))
      allocate(debug_cell(3*nphimax*(nrcrit+1),1))
      do i = 1,3*nphimax*(nrcrit+1)
        cell_list(i) = 0
        debug_cell(i,1) = 0
      enddo
    else
      allocate(cell_list(27))
      allocate(debug_cell(27,1))
      do i = 1,27
        cell_list(i) = 0
```

```
        debug_cell(i,1) = 0
      enddo
    endif

!  Check whether or not the neighbouring grid should be taken into account when building the neighbour list
! This is done by first including the actual cell, then all other cells are checked one by one

!    cell_list(1) = ncell0
!    cell_list(2) = nint(cell_number(nr,nphi,nz))
i = 1
    firstcellflag = 1

! ********
! special cases, nr =1 and nr=0 : add all cells surrounding the center + 1 ring more
    if (nr.le.nrcrit) then
      do k = 1,3
        nzk = nz+k-2
        if ((nzk.ge.1).and.(nzk.le.nzmax)) then
          do j = 1,nrcrit+1
            nrj = nr-1+j
            do l = 1,nphimax
              nphil = l
              cell_list(i) = cell_number(nrj,nphil,nzk)
              i = i+1
            enddo
          enddo
        endif
      enddo
      goto 20
    endif

    if (p(1).eq.0) then
      goto 20
    endif

! new NL - take into account ALL neighbouring cells.

!   other cases

    i = 1
do k = 1,3
      nzk = nz+k-2
      if ((nzk.ge.1).and.(nzk.le.nzmax)) then
        do j = 1,3
          nrj = nr+j-2
          if (nrj.le.nrmax) then
            do l = 1,3
              nphil = nphi+l-2
              if (nphil.gt.nphimax) then
                cell_list(i) = nint(cell_number(nrj,1,nzk))
                i = i+1
              endif
              if (nphil.eq.0) then
                cell_list(i) = nint(cell_number(nrj,nphimax,nzk))
                i =i+1
              endif
              if ((nphil.le.nphimax).and.(nphil.gt.0)) then
                cell_list(i) = nint(cell_number(nrj,nphil,nzk))
                i = i+1
              endif
            enddo
          endif
        enddo
      endif
    enddo
    if (debug.eq.1) then
      debug_cell(:,1) = cell_list
    endif

    goto 20 ! bypass everything below


! ********
!  outer loop. k = 1 corresponds to nz-1 which is 1 plane higher than the actual particle plane
! k = 2 to the actual particle plane
! and k = 3 to the lower plane

do k = 1,3
  if ((nz.eq.1).and.(firstcellflag.eq.1)) then  !special cases: nz=1, nr=nrmax
        firstcellflag = 0
        goto 10 ! skip entire loop
      endif

      if (k.eq.1) then
```

```
        nzj = nz-1
        zj = z_grid0
        dist = abs(zj-z) ! check up and down
        if (debug.eq.1) then
          debug_cell(m,1) = dist
!          debug_cell(m,2) = cell_number(nr,nphi,nzj)
          m = m+1
        endif
        if ((dist-tol).le.(0.5*d_i+0.5*d_0)) then
          cell_list(i) = nint(cell_number(nr,nphi,nzj))
          i = i+1
        endif
      endif
      if (k.eq.2) then
        nzj = nz
        zj = z
      endif
      if (k.eq.3) then
        if (nz.eq.nzmax) then
          goto 20
        endif
        nzj=nz+1
        zj = z_grid1
        dist = abs(zj-z) ! check up and down
        if (debug.eq.1) then
          debug_cell(m,1) = dist
!          debug_cell(m,2) = cell_number(nr,nphi,nzj)
          m = m+1
        endif
        if ((dist-tol).le.(0.5*d_i+0.5*d_0)) then
          if (nr.ne.1) then
            cell_list(i) = nint(cell_number(nr,nphi,nzj))
            i = i+1
          endif
        endif
      endif

! lower bounds
! ********

  do j = 1,3
       nrj = nr-2+j
        if ((nrj.eq.0).or.(nrj.eq.(nrmax+1))) then  ! skip since there is no neighbours here for obvious reasons
          goto 30
        endif
!        if ((nrj.eq.1).and.(nr.eq.1)) then  ! skip since they have already been added for nr=1
!          goto 30
!        endif
!        rgridj = nrj*dr
        if ((j.eq.1) .or.(j.eq.3)) then
          if (j.eq.1) then
            rgridj = (nr-1)*dr
          endif
          if (j.eq.3) then
            rgridj = nr*dr
          endif

          xj = rgridj*cos(phi_grid1)
          yj = rgridj*sin(phi_grid1)
          dist = sqrt((p(1)-xj)**2+(p(2)-yj)**2+(p(3)-zj)**2)
          if (debug.eq.1) then
            debug_cell(m,1) = dist
            if ((nphi-1).eq.0) then ! necessary since there is no nphi=0, rather nphi=0 corresponds to nphi=nphimax
!              debug_cell(m,2) = cell_number(nrj,nphimax,nzj)
            else
!              debug_cell(m,2) = cell_number(nrj,nphi-1,nzj)
            endif
            m = m+1
          endif
  if ((dist-tol).le.(0.5*d_i+0.5*d_0)) then
            if ((nphi-1).eq.0) then ! necessary since there is no nphi=0, rather nphi=0 corresponds to nphi=nphimax
              cell_list(i) = nint(cell_number(nrj,nphimax,nzj))
            else
              cell_list(i) = nint(cell_number(nrj,nphi-1,nzj))
            endif
            i = i+1
          endif
        else
          dist = sqrt((r*sin(abs(phi-phi_grid1)))**2+(p(3)-zj)**2)
          if (debug.eq.1) then
            debug_cell(m,1) = dist
            if ((nphi-1).eq.0) then
!              debug_cell(m,2) = cell_number(nrj,nphimax,nzj)
            else
```

```
!             debug_cell(m,2) = cell_number(nrj,nphi-1,nzj)
            endif
            m = m+1
          endif
          if ((dist-tol).le.(0.5*d_i+0.5*d_0)) then
            if ((nphi-1).eq.0) then
              cell_list(i) = nint(cell_number(nrj,nphimax,nzj))
            else
              cell_list(i) = nint(cell_number(nrj,nphi-1,nzj))
            endif
            i=i+1
          endif
        endif
30    enddo

!  ************
!   upper bounds (nphi=nphi_grid+1)
      do j =1,3
        nrj = nr-2+j
        if ((nrj.eq.0).or.(nrj.eq.(nrmax+1))) then ! skip since there is no neighbours here for obvious reasons
          goto 40
        endif
!        if ((nrj.eq.1).and.(nr.eq.1)) then ! skip since they have already been added for nr=1
!          goto 40
!        endif

!        rgridj = nrj*dr
        if ((j.eq.1) .or.(j.eq.3)) then
          if (j.eq.1) then
            rgridj = (nr-1)*dr
          endif
          if (j.eq.3) then
            rgridj = nr*dr
          endif
          xj = rgridj*cos(phi_grid0)
          yj = rgridj*sin(phi_grid0)
          dist = sqrt((p(1)-xj)**2+(p(2)-yj)**2+(p(3)-zj)**2)
          if (debug.eq.1) then
            debug_cell(m,1) = dist
            if (nphi.eq.nphimax) then ! necessary since there is no nphi=nphimax+1, rather nphi=nphimax+1 corresponds to nphi=1
!             debug_cell(m,2) = cell_number(nrj,1,nzj)
            else
!              debug_cell(m,2) = cell_number(nrj,nphi+1,nzj)
            endif
            m = m+1
          endif
  if ((dist-tol).le.(0.5*d_i+0.5*d_0)) then
            if (nphi.eq.nphimax) then ! necessary since there is no nphi=nphimax+1, rather nphi=nphimax+1 corresponds to nphi=1
            cell_list(i) = nint(cell_number(nrj,1,nzj))
            else
              cell_list(i) = nint(cell_number(nrj,nphi+1,nzj))
            endif
            i = i+1
          endif
        else
         dist = sqrt((r*sin(abs(phi-phi_grid0)))**2+(p(3)-zj)**2)
          if (debug.eq.1) then
            debug_cell(m,1) = dist
            if (nphi.eq.nphimax) then
!              debug_cell(m,2) = cell_number(nrj,1,nzj)
            else
!              debug_cell(m,2) = cell_number(nrj,nphi+1,nzj)
            endif
            m = m+1
          endif
      if ((dist-tol).le.(0.5*d_i+0.5*d_0)) then
            if (nphi.eq.nphimax) then
              cell_list(i) = nint(cell_number(nrj,1,nzj))
            else
              cell_list(i) = nint(cell_number(nrj,nphi+1,nzj))
            endif
            i = i+1
          endif
        endif
40    enddo
!   two last side cells
!   left
      dist = sqrt((r-r_grid0)**2+(p(3)-zj)**2)
      if (debug.eq.1) then
        debug_cell(m,1) = dist
!        debug_cell(m,2) = cell_number(nr-1,nphi,nzj)
        m = m+1
      endif
      if ((dist-tol).le.(0.5*d_i+0.5*d_0)) then
```

```
        if (nr.eq.1) then
          goto 10 ! if nr = 1 there is no "left" grid neighbour due to the construction of the grid, however
              ! there may a "right" neighbour grid
        endif
        cell_list(i) = nint(cell_number(nr-1,nphi,nzj))
        i = i+1
      endif
!     right
      dist = sqrt((r-r_grid1)**2+(p(3)-zj)**2)
      if ((debug.eq.1).and.(nr.ne.nrmax)) then
        debug_cell(m,1) = dist
!        debug_cell(m,2) = cell_number(nr+1,nphi,nzj)
        m = m+1
      endif
      if ((dist-tol).le.(0.5*d_i+0.5*d_0)) then
        if (nr.eq.nrmax) then
          goto 10 ! if nr = nrmax there is no "right" grid neighbour due to the construction of the grid, however
              ! there may be a "left" neighbour grid
        endif
        cell_list(i) = nint(cell_number(nr+1,nphi,nzj))
        i = i+1
      endif
10  enddo


20  call sort(i-1,cell_list) ! sorts the cell list from minimum to maximum number


! ***** Generate neighbour list
    numberCells = i-1
    iStart = 1
    particles = 0

    allocate (ipoint(numberCells))

    do j =1,numberCells ! calculate the number of particles in each grid to dimensionalize the neighbour_list matrix
      call nl_size(iStart,cell_list(j),particles,ipoint(j))
    enddo

    if (particles.gt.0) then
      empty_list = 0
      nlSize = particles
      deallocate (neighbour_list)
      allocate (neighbour_list(1:nlSize,3))
      particles = 1
      do j = 1,numberCells
        iStart = ipoint(j)
        call build_nl(iStart,cell_list(j),particles)
      enddo
    else
      empty_list = 1 ! the list is empty if the number of particles = 0
    endif
    deallocate (ipoint)
    deallocate (cell_list)
    empty_old = empty_list
    ncellold = ncell
end

!  This subroutine determines the required size of the neighbour_list matrix. The number of particles in each grid is counted.

subroutine nl_size(i,cell,p,ipoint)
use variables_deposition
    implicit none
    integer*4 i,cell,p,ipoint,iflag

    iflag = 0
    ipoint = 0

    do
      if (nint(part_pos(i,1)).eq.cell) then
        p = p+1
        if (iflag.ne.1) then
          ipoint = i
          iflag = 1
        endif
      endif
      if ((nint(part_pos(i,1)).gt.cell).or.(i.eq.size(part_pos,1))) then
        exit
      endif
      i = i+1
    enddo

end
```

```
! This subroutine builds up the neighbour list from the part_pos matrix. It is required that the nl_bounds subroutine is called
! prior to this subroutine

subroutine build_nl(i,cell,p) !input iStart, cell number, particle count
use variables_deposition
    implicit none
    integer*4 i,k
    integer*4 cell,p

    if (i.eq.0) then
      return
    endif

    do
      if (nint(part_pos(i,1)).eq.cell) then
        do k =1,3
          neighbour_list(p,k) = part_pos(i,k+1)
        enddo
        p = p+1
      endif
      if ((nint(part_pos(i,1)).gt.cell).or.(i.eq.size(part_pos,1))) then
        exit
      endif
      i = i+1
    enddo

end
```

F.3.5 Sort_array.f90

```
subroutine sort(sizeArray,array)
implicit none
    integer*4 ifflag,ArrayFront,ArrayActual,i,sizeArray
    integer*4 array(sizeArray)

      i = 1
      if (sizeArray.eq.1) then
        return
      else
        do
          if (array(i+1).lt.array(i)) then
            arrayFront = array(i+1)
            arrayActual = array(i)

            array(i+1) = arrayActual
            array(i) = ArrayFront
            i = 0
          endif
          i = i+1
          if (i.eq.sizeArray) then
            exit
          endif
        enddo
      endif
end

subroutine sort_real(i,kplus,sizeArray,array,stopFlag)
use variables_deposition
    implicit none
    integer*4 i,sizeArray,iStart,stopFlag,k,kplus
    real*8 array(sizeArray,3),arrayFront,ArrayActual

    iStart = i-1
    k = kplus-nphimax*nrmax

    if (sizeArray.eq.1) then
      return
    else
      do
        if ((array(i+1,3).lt.kplus).and.(array(i+1,3).ge.k)) then
          if (array(i+1,2).lt.array(i,2)) then
            arrayFront = array(i+1,2)
            arrayActual = array(i,2)

            array(i+1,2) = arrayActual
            array(i,2) = ArrayFront
            i = iStart
          endif
          if ((i+1).ge.sizeArray) then
            stopFlag = 1
```

```
          exit
        endif
      else
        i = i+1
        exit
      endif
      i = i+1
    enddo
  endif
end
```

F.3.6 Calculate_collisions.f90

```
subroutine initial_boundary_check(coll)
use variables_deposition
    implicit none
    real*8 dewall,tol
    integer*4 coll
!    tol = 1d-16
tol = 0d0
    dewall = sqrt(r0(1)**2+r0(2)**2)

    if ((dewall-tol).ge.(radius-0.5*d_0)) then
      coll = 1
    else
  coll = 0
    endif
end

subroutine boundary_check(p,coll)
use variables_deposition
    implicit none
    real*8 dewall,p(3)
    real*8 tol
    integer*4 coll,grid_pos(3)

!    tol = 1d-10
tol = 0d0
    dewall = sqrt(p(1)**2+p(2)**2)
    call check_grid(p,grid_pos)

    if ((grid_pos(3).le.skipIncludingCell).and.(skipfirst.eq.1).and.(filmdeposition.ne.1)) then
      if ((dewall-tol).gt.(radius-0.5d0*d_0)) then
        coll = 1
      else
    coll = 0
      endif
    else
      coll = 0
      if (filmdeposition.eq.1) then
        dewall = zmax-p(3)
        if (dewall .le.(0.5d0*d_0)) then
          coll = 1
        else
          coll = 0
        endif
      endif
    endif
end

! ***** This subroutine is used in combination with the drop_particle() subroutine. It checks if the selected coordinates
! ***** are allowed or not. If the distance between the neighbouring particles and the newly dropped particle mean that
! ***** the two are overlapping, the subroutine returns with 1.

subroutine initial_collision_check(pdep,coll)
use variables_deposition
    implicit none
    real*8 dei1,pdep(3)
    integer*4 coll

dei1 = sqrt((r0(1)-pdep(1))**2+(r0(2)-pdep(2))**2+(r0(3)-pdep(3))**2)
    if ((dei1) .le. (0.5*(d_i+d_0))) then
      coll = 1
    else
      coll = 0
      if (abs(dei1-(0.5*(d_i+d_0))).lt.1d-16) then
        write(*,*) 'machine accuaracy questioned'
      endif
    endif
end

! ***** This subroutine checks if the moving particle after movement collides with a particle in its neighbour list.
```

```
! ***** Returns 1 if this is true, 0 if not.

subroutine collision_check(empty_list,pdep,coll)
use variables_deposition
    implicit none
    real*8 minDist,pdep(3),a,b,c,d
    real*8 xex0(3),xexdep(3),x0xdep(3),crossprod(3),veclength
    real*8 tmark,tmax,tol,endDist,tc,tc1,tc2,dot_product
    integer*4 coll,empty_list
    if (empty_list.eq.1) then
      coll = 0
      return
    endif

! ----- http://mathworld.wolfram.com/Point-LineDistance3-Dimensional.html - distance between line (r0->re) and point (pdep)
    tol = 1d-10
    xex0 = re-r0
    x0xdep = r0-pdep
    xexdep = re-pdep

    call crossp(xex0,x0xdep,crossprod)

    minDist = veclength(crossprod)/veclength(xex0) ! min distance
    if ((minDist+tol).gt.(d_0)) then
      coll = 0
      return
    endif

    a = veclength(xex0)**2d0
    b = 2d0*dot_product(xex0,x0xdep)
    c = veclength(x0xdep)**2d0-d_0**2d0
    d = b**2-4d0*a*c

    tc1 = -1d0*(b+dsqrt(d))/(2d0*a)
    tc2 = -1d0*(b-dsqrt(d))/(2d0*a)

    if (tc1.lt.tc2) then
      tc = tc1
    else
      tc = tc2
    endif

    if ((tc.le.1d0).and.(tc.gt.0d0)) then
      coll = 1
      return
    else
      coll = 0
      return
    endif
end

! ***** This subroutine checks if a collision between the wall and a particle takes place and returns an integer corresponding
! ***** to true or false.

subroutine wall_collision_check(p,wall_coll)
use variables_deposition
    implicit none
    real*8 dewall,p(3),tol
    integer*4 wall_coll,grid_pos(3)

!    tol = 1d-16
    tol = 0d0
    dewall = sqrt(p(1)**2+p(2)**2)

    if ((dewall-tol).ge.(radius-0.5d0*d_0)) then
      wall_coll = 1
    else
      wall_coll = 0
    endif

end

! ***** This subroutine calculates the collision coordinates between the depositing particle and a particle in its neighbour list
! ***** Since multiple collisions are possible (though not highly probable), the routine also returns the distance
! ***** travelled by the depositing particle to the colliding particle. To compare, the smallest distance at which the depositing
! ***** particle has to travel to deposit, will correspond to the particle at which it collides first.

subroutine collision_coordinates(pdep,rcoll,dist)
use variables_deposition
    implicit none
real*8 a1,a2,b1,b2,b3
    real*8 rcoll(3),rc1(3),rc2(3) ! rc=(xc,yc,zc)
    real*8 d0i1,d0i2,pdep(3),dist,dei1,xex0(3),veclength,tmark
    integer*4 i
```

```
    a1 = r0(2)*re(1)+pdep(2)*r0(1)-r0(1)*re(2)-pdep(2)*re(1)
    a2 = r0(3)*re(1)+pdep(3)*r0(1)-r0(1)*re(3)-pdep(3)*re(1)

    b1 = 1d0 + ((re(2)-r0(2))/(re(1)-r0(1)))**2 + ((re(3)-r0(3))/(re(1)-r0(1)))**2
    b2 = (2d0*(re(3)-r0(3))*a2+2d0*(re(2)-r0(2))*a1-2d0*pdep(1)*(re(1)-r0(1))**2)/((re(1)-r0(1))**2)
    b3 = (pdep(1)**2*(re(1)-r0(1))**2+a1**2+a2**2-0.25d0*(d_0+d_i)**2*(re(1)-r0(1))**2)/((re(1)-r0(1))**2)
    dei1 = sqrt((re(1)-pdep(1))**2+(re(2)-pdep(2))**2+(re(3)-pdep(3))**2)

    rc1(1) = (-b2+dsqrt(b2**2-4d0*b1*b3))/(2d0*b1)
    rc2(1) = (-b2-dsqrt(b2**2-4d0*b1*b3))/(2d0*b1)

    do i=2,3
      rc1(i) = (re(i)-r0(i))/(re(1)-r0(1))*(rc1(1)-r0(1))+r0(i)
      rc2(i) = (re(i)-r0(i))/(re(1)-r0(1))*(rc2(1)-r0(1))+r0(i)
    enddo

    d0i1 = dsqrt((rc1(1)-r0(1))**2+(rc1(2)-r0(2))**2+(rc1(3)-r0(3))**2)
    d0i2 = dsqrt((rc2(1)-r0(1))**2+(rc2(2)-r0(2))**2+(rc2(3)-r0(3))**2)

    if ((d0i1).lt.(d0i2)) then
      rcoll = rc1
    else
      rcoll = rc2
    endif
    dist = dsqrt((r0(1)-rcoll(1))**2+(r0(2)-rcoll(2))**2+(r0(3)-rcoll(3))**2) ! the distance moved is calculated to see which particle is hit first

    xex0 = re-r0  ! necessary if multible collisions can occur. Minimum distance => particle hit
    tmark = dist/veclength(xex0)
    if (tmark.gt.1d0) then
      write(*,*) 'something is wrong'
      write(*,*) tmark
!      read(*,*)
    else
 !     write(*,*) tmark
    endif
end

! ***** This subroutine calculates the collision coordinates with a particle and a cylinder of radius "radius"
! ***** No input. Output is the collision coordinates returned in vector rcoll in which rcoll=(xc,yc,zc)

subroutine wall_collision_coordinates(coll,rcoll)
use variables_deposition
implicit none
real*8 c1,c2,c3
    real*8 xc1,xc2,yc1,yc2
    reaL*8 d1,d2,phic,phie,phi0,xc,yc,zc,r_0,r_e
    real*8 a,b,c,tcplus,tcminus,tc
    real*8 xe,x0,ye,y0,ze,z0
    real*8 rcoll(3),r0Cyll(3),reCyll(3),rtest,tol
    integer*4 coll
    tol = 1d-10
if (coll.eq.0) then
      return
    endif

    xe = re(1)
    x0 = r0(1)
    ye = re(2)
    y0 = r0(2)
    ze = re(3)
    z0 = r0(3)

    if (filmdeposition.eq.1) then ! calculate collision between plane and line
      tc = (zmax-0.5d0*d_0-z0)/(ze-z0)
      xc = (xe-x0)*tc+x0
      yc = (ye-y0)*tc+y0
      zc = zmax-0.5d0*d_0
      rc(1) = xc
      rc(2) = yc
      rc(3) = zc
      return
    endif

    a = (xe-x0)**2+(ye-y0)**2
    b = 2d0*(xe-x0)*x0+2d0*(ye-y0)*y0
    if (geometry.eq.0) then
      c = x0**2+y0**2-(radius-0.00001*d_0pseudo)**2
    else
      c = x0**2+y0**2-(radius-0.5d0*d_0)**2
    endif

    tcplus = (-b+sqrt(b**2-4d0*a*c))/(2d0*a)
```

```
    tcminus =(-b-sqrt(b**2-4d0*a*c))/(2d0*a)

    if (((tcplus-tol).le.1d0).and.((tcplus+tol).ge.0d0)) then
      tc = tcplus
    else
      tc = tcminus
    endif

    if ((((tcplus-tol).gt.1d0).or.((tcplus+tol).lt.0)).and.(((tcminus-tol).gt.1d0).or.((tcminus+tol).lt.0))) then
      write(*,*) 'error: ', tcplus,tcminus
!      read(*,*)
    endif
    xc = x0+(xe-x0)*tc
    yc = y0+(ye-y0)*tc
    zc = z0+(ze-z0)*tc

!    call cylindrical_coordinates(r0,r0Cyll)
!    call cylindrical_coordinates(re,reCyll)
!
!    r_0 = r0Cyll(1)
!    r_e = reCyll(1)
! phie = reCyll(2)
!    phi0 = r0Cyll(2)

!    if (geometry.eq.0) then
!      phic = phi0+(phie-phi0)*(radius-0.0001*d_0pseudo-r_0)/(r_e-r_0)
!      xc = (radius-0.0001*d_0pseudo)*cos(phic)
!      yc = (radius-0.0001*d_0pseudo)*sin(phic)
!    zc = r0(3)+(re(3)-r0(3))*(radius-0.0001*d_0pseudo-r_0)/(r_e-r_0)
!    else
!      phic = phi0+(phie-phi0)*(radius-0.5d0*d_0-r_0)/(r_e-r_0)
!      xc = (radius-0.5d0*d_0)*cos(phic)
!      yc = (radius-0.5d0*d_0)*sin(phic)
!      zc = r0(3)+(re(3)-r0(3))*(radius-0.5d0*d_0-r_0)/(r_e-r_0)
!    endif

!    if (debug.eq.1) then
!      write(20,*) '# Phi0, r_0'
!      write(20,*) phi0, r_0
!      write(20,*) '# PhiE, r_e'
!      write(20,*) phie,r_e
!      write(20,*) '# PhiC, r_c'
!      write(20,*) phic,radius-0.5d0*d_0
!    endif
    rtest = dsqrt(xc**2+yc**2)
    if ((rtest-tol).gt.(radius-0.5d0*d_0)) then
      write(*,*) 'nu bliver jeg snart vred: ', rtest, radius-0.5d0*d_0
!      read(*,*)
    endif
    rcoll(1) = xc
    rcoll(2) = yc
    rcoll(3) = zc


end

! ***** The below subroutine calculates the distance between _all_ deposited particles. If any distance is less than 1 diameter,
! ***** the subroutine returns an error message, which is used in debugging to check if any particles overlap (even though
! ***** there should be no possibility that they can do this).

subroutine brute_check(buggedPart,ncelli,ncellj,r1,r2)
use variables_deposition
    implicit none
    real*8 xi,yi,zi,xj,yj,zj,dist,tol
    real*8 r1(3),r2(3)
    integer*4 i,j,buggedPart,ncelli,ncellj

    buggedPart = 0
    tol = 1d-16
    do i = 1,size(part_pos,1)
      xi = part_pos(i,2)
      yi = part_pos(i,3)
      zi = part_pos(i,4)
      ncelli = nint(part_pos(i,1))
      do j =1,size(part_pos,1)
        xj = part_pos(j,2)
        yj = part_pos(j,3)
        zj = part_pos(j,4)
        ncellj = nint(part_pos(j,1))


        if (i.eq.j) then
          dist = 1d8
        else
```

```
          dist = sqrt((xi-xj)**2+(yi-yj)**2+(zi-zj)**2)
        endif
        if ((dist+tol).lt.(0.5*(d_i+d_0))) then
          r1(1) = xi
          r1(2) = yi
          r1(3) = zi

          r2(1) = xj
          r2(2) = yj
          r2(3) = zj

          buggedPart = buggedPart+1
          write(*,*) dist
          return
        endif
      enddo
    enddo
!    write(*,*) buggedPart
end

subroutine debug_collisions(numcoll)
use variables_deposition
    implicit none
    integer*4 parti,partj,overlapping,empty_flag,i,grid_end(3)
    integer*4 coll,numcoll

    real*8 r1(3),r2(3)

! ----- For debugging
    call check_grid(re,grid_end)

    if ((geometry.eq.0).or.(brutecheck.eq.0)) then
      overlapping = 0
    else
     call brute_check(overlapping,parti,partj,r1,r2)
    endif
    if (overlapping.gt.0) then
      debug = 1
      write(*,*) 'Number of collions between particles', numcoll-1
      write(*,*) 'Particles are overlapping!'
      write(*,*) 'Moving particle in cell', parti
      write(*,*) 'is touching particle in cell', partj
     write(*,*) 'Program will terminate'
      call write_line_to_file()
      write(*,*) rc
!      read(*,*)
      call check_nl(re,grid_end,empty_flag)
      call write_neighbourlist(r1,r2)
      do i =1,size(neighbour_list,1)
        call collision_check(empty_flag,neighbour_list(i,:),coll)
      enddo
      write(20,*) 'Collision coordinates calculated for particle-particle collision'
      stop_program = 1
    endif
! ----- For debugging END
end
```

F.3.7 Store_particle.f90

```
subroutine store_particle()
use variables_deposition
    implicit none
    real*8, dimension(:,:), allocatable :: part_pos_temp(:,:)
    real*8 cell_number
    integer*4 i,j,k,n,m,l,grid_pos(3)
    integer*4 nr,nphi,nz,ncell,iFlag

    call check_grid(rc,grid_pos) ! get grid position

    nr = grid_pos(1)
    nphi = grid_pos(2)
    nz = grid_pos(3)

    if (rc(1)*0d0.ne.0d0) then !NaN check
      dropped = dropped+1
      return
    endif

    if ((nr.gt.nrmax).and.((skipfirst.eq.1).or.(filmdeposition.eq.1))) then
      lost_side = lost_side+1
      return
    endif
```

```
    if ((nz.eq.1).and.((skipfirst.eq.1).or.(filmdeposition.eq.1))) then
      firstcelldep = firstcelldep+1
      stop_program = 1
      write(*,*) 'Program terminated due to particle deposition in first cell' ! MODIFY
    endif

    ncell = cell_number(nr,nphi,nz) ! calculate corresponding cell number

i = 1
    m = 0
    k = size(part_pos,1)+1
    allocate (part_pos_temp(k,4))

    iFlag = 0
do ! loop : count up to the first place where i is greater than the actual cell number
      m = m+1
      i = part_pos(m,1)
      if (i.gt.ncell) then
        iFlag = 1
        exit
      endif
      if (m.eq.(k-1)) then
exit
      endif
    enddo
! ********
! Case I: first cell number in part_pos is larger than the actual cell number, therefor we insert the particle at the first position
! in part_pos. e.g. (2,2,3,3,4) with actual cell number (1).

    if ((iFlag.eq.1) .and. (m.eq.1)) then
      part_pos_temp(1,1) = ncell
      part_pos_temp(1,2) = rc(1)
      part_pos_temp(1,3) = rc(2)
      part_pos_temp(1,4) = rc(3)
      do j =2,k
        do l = 1,4
          part_pos_temp(j,l) = part_pos(j-1,l)
        enddo
      enddo
    endif

! Case II: actual cell number is between the first and the last cell number in part_pos. The depositing particle is inserted
!  in the correct position of part_pos sorted by cell number, e.g. (3,3,4,5,7) with actual cell number (6).

    if ((iFlag.eq.1).and.(m.gt.1)) then
      do l =1,m-1
        do n = 1,4
          part_pos_temp(l,n) = part_pos(l,n)
        enddo
      enddo
      part_pos_temp(m,1) = ncell
      part_pos_temp(m,2) = rc(1)
      part_pos_temp(m,3) = rc(2)
      part_pos_temp(m,4) = rc(3)
      do j = m+1,k
        do l = 1,4
          part_pos_temp(j,l) = part_pos(j-1,l)
        enddo
      enddo
    endif
! Case III: actual cell number is larger than the largest one in the existing part_pos matrix. The depositing particle is
!   then inserted as the last particle, e.g. (3,4,5,6,6) with actual cell number (8).

    if ((iFlag.ne.1).and.(m.eq.(k-1))) then
      do l = 1,m
        do n = 1,4
          part_pos_temp(l,n) = part_pos(l,n)
        enddo
      enddo
      part_pos_temp(m+1,1) = ncell
      part_pos_temp(m+1,2) = rc(1)
      part_pos_temp(m+1,3) = rc(2)
      part_pos_temp(m+1,4) = rc(3)
    endif

    deallocate (part_pos)
    allocate (part_pos(k,4))
    part_pos = part_pos_temp
    deallocate (part_pos_temp)

end

subroutine store_first_particle()
```

```
use variables_deposition
    implicit none
    integer*4 nr,nphi,nz,grid_pos(3),ncell
    real*8 cell_number

    call check_grid(rc,grid_pos) ! get grid position
    nr = grid_pos(1)
    nphi = grid_pos(2)
    nz = grid_pos(3)

    ncell = cell_number(nr,nphi,nz) ! calculate corresponding cell number

    part_pos(1,1) = ncell
    part_pos(1,2) = rc(1)
    part_pos(1,3) = rc(2)
    part_pos(1,4) = rc(3)
end
```

F.3.8 Check_grid.f90

```
subroutine check_grid(p,grid_pos)
use variables_deposition
    implicit none
    real*8 r,phi,z
    real*8 p(3)
    integer*4 grid_pos(3),nr,nphi,nz

    r = sqrt(p(1)**2+p(2)**2)
    if (p(1).eq.0) then
      phi = 0
    else
      if ((p(1).gt.0).and.(p(2).gt.0)) then
        phi = atan(p(2)/p(1))
      endif
      if ((p(1).lt.0).and.(p(2).gt.0)) then
        phi = pi-abs(atan(p(2)/p(1)))
      endif
      if ((p(1).lt.0).and.(p(2).lt.0)) then
        phi = pi+abs(atan(p(2)/p(1)))
      endif
      if ((p(1).gt.0).and.(p(2).lt.0)) then
        phi = 2*pi-abs(atan(p(2)/p(1)))
      endif
      if (phi.lt.0) then
        write(*,*) 'dups'
      endif
    endif
    z=p(3)

    nr = int(r/dr)+1
    nphi = int(phi/dphi)+1
! ---- Modification 0<z<zfirst - to obtain truly random motion
    if (z.lt.dzfirst) then
      nz = 1
    else
      nz = int((z-dzfirst)/dz)+2
    endif

    grid_pos(1) = nr
    grid_pos(2) = nphi
    grid_pos(3) = nz
end
```

F.3.9 Cylindrical_coordinates.f90

```
subroutine cylindrical_coordinates(pCart,pCyl)
use variables_deposition
implicit none
    real*8 r,phi
    real*8 pCart(3),pCyl(3)

    r = sqrt(pCart(1)**2+pCart(2)**2)
    if (pCart(1).eq.0) then
      phi = 0
    else
      if ((pCart(1).gt.0).and.(pCart(2).gt.0)) then
        phi = atan(pCart(2)/pCart(1))
      endif
      if ((pCart(1).lt.0).and.(pCart(2).gt.0)) then
        phi = pi-abs(atan(pCart(2)/pCart(1)))
```

```
      endif
      if ((pCart(1).lt.0).and.(pCart(2).lt.0)) then
        phi = pi+abs(atan(pCart(2)/pCart(1)))
      endif
      if ((pCart(1).gt.0).and.(pCart(2).lt.0)) then
        phi = 2*pi-abs(atan(pCart(2)/pCart(1)))
      endif
      if (phi.lt.0) then
        write(*,*) 'dups'
      endif
    endif

    pCyl(1) = r
    pCyl(2) = phi
    pCyl(3) = pCart(3)
    return
end
```

F.3.10 Input_output.f90

```
subroutine write_header_to_file()
use variables_deposition
    implicit none
    integer*4 unit,i

    do i = 0,8
      unit = i*5+5
      if ((continuation.eq.1)) then
        return
      endif

      write(unit,*) '# '
      write(unit,*) '# Particle dynamics simulation: deposition in cylindrical capillaries'
      write(unit,*) '# Program by Tobias D. Elmoe'
      write(unit,*) '# '
      write(unit,*) '# File created:',creation_date, '@', creation_time
      write(unit,*) '# '

      if (unit.eq.5) then
        write(5,*) '# Inserted, time at deposition [s]'
      endif
      if (unit.eq.10) then
        write(10,*) '# Ncell, X, Y, Z, time at deposition [s]'
      endif
      if (unit.eq.15) then
        write(15,*) '# Ncell, X, Y, Z'
      endif
      if (unit.eq.20) then
        write(20,*) '# [!Ignore if below is empty!]'
        write(20,*) '# ---- ! FILE GENERATED DUE TO ERROR IN PROGRAM ! ---- '
        write(20,*) '# The actual neighbour list at the time of crash is seen below'
      endif
      if (unit.eq.30) then
        write(30,*) '#  [!Ignore if below is empty!]'
        write(30,*) '# ---- ! FILE GENERATED DUE TO ERROR IN PROGRAM ! ---- '
        write(30,*) '# distance from end cell to given cell is seen below'
        write(30,*) '# distance [m], Ncell'
      endif
      if (unit.eq.40) then
        write(40,*) '# mu [ =4*Dp*z/(pi*(d_0)*u_avr) ], P [ = penetration ]'
      endif
  if (unit.eq.45) then
        write(45,*) '# Neighbourlist time, Integrator time [s]'
      endif
10 enddo
end

subroutine write_specifications()
use variables_deposition
    implicit none
    write(25,10) Pe
    write(25,50) ugasmax/2
    write(25,100) d_0
    write(25,150) radius
    write(25,200) nrmax,nphimax,nzmax
    write(25,250) dropped_max
    write(25,300) geometry
    write(25,325) filmdeposition
    write(25,350) disp
10  format('# Peclet number : ', F8.4)
```

```
50  format('# Average gas velocity                         : ',F8.4,1X,'[m/s]')
100 format('# Particle diameter                            : ',E16.6,1X,'[m]')
150 format('# Tube radius                                  : ',E16.6,1X,'[m]')
200 format('# Grid division (r,phi,z)                      : ',3I6.4)
250 format('# Maximum allowable number of particles dropped : ', I16.8)
300 format('# Geometry included (1: Yes, 0: No          : ', I6.3)
325 format('# Film deposition (1): yes, 0: No              : ', I6.3)
350 format('# Percentage of diameter moved each time step   : ', F6.3)
end

subroutine write_summery()
use variables_deposition
    implicit none


100 format('Number of particles deposited totally                 : ',I16.8)
200 format('Number of particles not deposited (side, bottom)      : ',2I16.8)
250 format('Number of particles deposited in top-cell (0 if geo=0) : ', I16.8)
300 format('Total deposition time                                 : ',F16.8,'[s]')

    write(35,100) size(part_pos,1)
    write(35,200) lost_side,lost_bottom
    write(35,250) firstcelldep
    write(35,300) actual_time
end


subroutine write_to_file()
use variables_deposition
implicit none
    integer*4 i

100 format(F10.1,',',E23.16,',',E23.16,',',E23.16)

    do i =1,size(part_pos,1)
      write(15,100) part_pos(i,1), part_pos(i,2),part_pos(i,3),part_pos(i,4)
    enddo
end



subroutine write_line_to_file()
use variables_deposition
    implicit none
    integer*4 grid_pos(3),nr,nphi,nz
    integer*4 hour1,hour0,minute1,minute0,seconds1,seconds0
    real*8 cell,cell_number,startTime,runTime,realTime,ms1,ms0
    character*10 real_time,dum1

100 format(F10.1,',',E16.6,',',E16.6,',',E16.6,',',E16.6)
200 format(I10.1,',',I6.1,',',I6.1,',',E16.6)
300 format(E16.8,2X,E16.8,2X,E16.8)

    call check_grid(rc,grid_pos)

    nr = grid_pos(1)
    nphi = grid_pos(2)
    nz = grid_pos(3)

    cell = cell_number(nr,nphi,nz)
    call date_and_time(dum1,real_time)
    read(real_time,*) realTime

    hour1 = int(realTime/10000)
    minute1 = int((realTime-dfloat(hour1)*10000)/100)
    seconds1 = int(realTime)-(hour1*10000+minute1*100)
    ms1 = (realTime-dfloat(int(realTime)))

    read(creation_time,*) startTime
    hour0 = int(startTime/10000)
    minute0 = int((startTime-dfloat(hour0)*10000)/100)
    seconds0 = int(startTime)-(hour1*10000+minute0*100)
    ms0 = (startTime-dfloat(int(startTime)))

    if (hour1.lt.hour0) then
      hour1 = hour1+24
    endif
    runTime = dfloat(hour1-hour0)*3600d0+dfloat(minute1-minute0)*60d0+dfloat(seconds1-seconds0)+ms1-ms0

    if (nr.le.nrmax) then
      write(45,300) intTime_nl,intTime_int,runTime
      write(10,100) cell,rc(1),rc(2),rc(3),actual_time
      write(5,*) actual_time, inserted
! ***** write to screen every 10 particles deposited
```

```
      if (mod(size(part_pos,1),10).eq.0) then
        write(*,*) size(part_pos,1),actual_time,real_time,lost_bottom
!        write(*,300)  intTime_nl,intTime_int
      endif
    endif

end

subroutine write_neighbourlist(r1,r2)
use variables_deposition
    implicit none
    integer*4 nl_size,i,grid(3),iend
    real*8 ncell,cell_number
    real*8 distance,xcxdep(3),veclength
    real*8 r1(3),r2(3)

    nl_size = size(neighbour_list,1)


100 format(F10.1,',',E16.6,',',E16.6,',',E16.6)
150 format(F10.1,',',E16.6,',',E16.6,',',E16.6,',',E16.6)
200 format(E16.6,2X,F6.1)

    do i = 1,nl_size
      xcxdep = rc-neighbour_list(i,:)
      distance = veclength(xcxdep)
      call check_grid(neighbour_list(i,:),grid)
      ncell = cell_number(grid(1),grid(2),grid(3))
      write(20,150) ncell, neighbour_list(i,1),neighbour_list(i,2),neighbour_list(i,3),distance
    enddo
    call check_grid(rc,grid)
    ncell = cell_number(grid(1),grid(2),grid(3))
    write(20,*) '# ------'
    write(20,*) '# Collision coordinates and cell number (nr,nphi,nz)',grid(1),grid(2),grid(3)
    write(20,100) ncell,rc(1),rc(2),rc(3)
    call check_grid(r0,grid)
    write(20,*) '# Start coordinates and cell number (nr,nphi,nz)', grid(1),grid(2),grid(3)
    ncell = cell_number(grid(1),grid(2),grid(3))
    write(20,100) ncell,r0(1),r0(2),r0(3)
    call check_grid(re,grid)
    write(20,*) '# End coordinates and cell number (nr,nphi,nz)', grid(1),grid(2),grid(3)
    ncell = cell_number(grid(1),grid(2),grid(3))
    write(20,100) ncell,re(1),re(2),re(3)
    call check_grid(r1,grid)
    ncell = cell_number(grid(1),grid(2),grid(3))
    write(20,*) '# Overlapping particle 1, coordinates & cell number'
    write(20,*) r1, ncell
    call check_grid(r2,grid)
    ncell = cell_number(grid(1),grid(2),grid(3))
    write(20,*) '# Overlapping particle 2, coordinates & cell number'
    write(20,*) r2, ncell

    iend = size(debug_cell,1)

    do i = 1,iend
      write(30,200) debug_cell(i,1)
    enddo

end

subroutine write_penetration_list()
use variables_deposition
    implicit none
real*8, dimension(:,:), allocatable :: temp(:,:)
    integer*4 i,k,kplus,stopFlag

100 format(E16.8,2X,F16.8)

    allocate (temp(size(part_pos,1),2))

    do i = 1,size(part_pos,1)
      temp(i,1) = 0
      temp(i,2) = part_pos(i,4) ! copy z-column
!      temp(i,3) = part_pos(i,1) ! copy cell number column
    enddo

    k = size(part_pos,1)
    call piksrt(k,temp(:,2))
    do i =1,size(part_pos,1)
      temp(k-i+1,1) = real(i)
      temp(i,2) = 4d0*Dp*temp(i,2)/(pi*(2d0*radius)**2*ugasmax*0.5d0)
    enddo
    do i = 1,size(part_pos,1)
      temp(i,1) = temp(i,1)/k
```

```
      write(40,100) temp(i,2), temp(i,1)
    enddo
end

subroutine write_to_temp()
use variables_deposition
    implicit none
    INTEGER, DIMENSION(:), ALLOCATABLE :: seed
    integer :: n,i

    CALL RANDOM_SEED(size = n)
    ALLOCATE(seed(n))

100 format(E23.16) ! double precision

    write(50,*) '# File ended due to time limit (1) yes, (0) no'
    write(50,*) timeLimitReached
    write(50,*) '# Original creation date'
    write(50,*) original_creation_date
    write(50,*) '# Original creation time'
    write(50,*) original_creation_time
    write(50,*) '# Actual time'
    write(50,100) actual_time
    write(50,*) '# Size seedValue'
    write(50,*) n
    write(50,*) '# Seed value'
    call random_seed(GET = seed) !gets information on the actual seed value
    do i =1,n
      write(50,*) seed(i)
    enddo
    write(50,*) '# Size of part_pos matrix'
    write(50,*) size(part_pos,1)
    write(50,*) '# Velocity vector for langevin dynamics, X,Y,Z'
    write(50,100) velocity(1)
    write(50,100) velocity(2)
    write(50,100) velocity(3)
    write(50,*) '# Position, X,Y,Z'
    write(50,100) re(1)
    write(50,100) re(2)
    write(50,100) re(3)
    write(50,*) '# Previous position, X,Y,Z'
    write(50,100) r0(1)
    write(50,100) r0(2)
    write(50,100) r0(3)
    write(50,*) '# inserted'
    write(50,*) inserted
    write(50,*) '# lost_side, lost_bottom'
    write(50,*) lost_side,lost_bottom
    write(50,*) '# First cell deposited'
    write(50,*) firstcelldep
    write(50,*) '# Dropped'
    write(50,*) dropped
    write(50,*) '# Particle still moving? (1) yes (0) no'
    call number_balance()
    if (balance.ne.0) then
      write(50,*) 1
    else
      write(50,*) 0
    endif
end


subroutine read_from_file()
use variables_deposition
    implicit none
    character*40 trash
    integer*4 sizePart_pos
    INTEGER, DIMENSION(:), ALLOCATABLE :: seedValue
    integer :: n,i




    read(50,*) trash
    read(50,*) original_creation_date
    read(50,*) trash
    read(50,*) original_creation_time
    read(50,*) trash
    read(50,*) actual_time
    read(50,*) trash
    read(50,*) n
    read(50,*) trash
    ALLOCATE(seedValue(n))
do i =1,n
```

```
!    do i =1,5
!      read(45,*) seedValue(1+(i-1)*6),seedValue(2+(i-1)*6),seedValue(3+(i-1)*6),seedValue(4+(i-1)*6),seedValue(5+(i-1)*6) &
!                 ,seedValue(6+(i-1)*6)
      read(50,*) seedValue(i)
    enddo

!    read(50,*) seedValue(31),seedValue(32),seedValue(33),seedValue(34)

    CALL RANDOM_SEED(PUT = seedValue)

    read(50,*) trash
    read(50,*) sizePart_pos
    read(50,*) trash
    read(50,*) velocity(1)
    read(50,*) velocity(2)
    read(50,*) velocity(3)
    read(50,*) trash
    read(50,*) re(1)
    read(50,*) re(2)
    read(50,*) re(3)
    read(50,*) trash
    read(50,*) r0(1)
    read(50,*) r0(2)
    read(50,*) r0(3)
    read(50,*) trash
    read(50,*) inserted
    read(50,*) trash
    read(50,*) lost_side,lost_bottom
    read(50,*) trash
    read(50,*) firstcelldep
    read(50,*) trash
    read(50,*) dropped
    read(50,*) trash
    read(50,*) moving


    deallocate (part_pos)
    allocate (part_pos(sizePart_pos,4))

!  read past header
    do i=1,6
      read(5,*) trash
      read(10,*) trash
      read(15,*) trash
  read(20,*) trash
      read(25,*) trash
      read(30,*) trash
      read(35,*) trash
      read(40,*) trash
      read(45,*) trash
    enddo

! for various files read past several lines
read(5,*) trash
    read(10,*) trash
    read(15,*) trash
    read(30,*) trash
    read(40,*) trash
    read(45,*) trash

    do i =1,3
      read(20,*) trash
      read(30,*) trash
    enddo

!  save values from file into matrix

    do i = 1,sizePart_pos
      read(5,*) trash ! moves the writing point to the last position of the file
      read(10,*) trash
      read(15,*) part_pos(i,1), part_pos(i,2), part_pos(i,3), part_pos(i,4)
      read(45,*) trash
    enddo

    close(15)
    close(50)


    open(unit=15,file='deposited_particles-srt-by-cell.dat')
    open(unit=50,file='temp.dat')

    do i =1,7
      read(15,*) trash
    enddo
```

```
end

subroutine read_input()
use variables_deposition
    implicit none
    character*40 trash
    real*8 test,cap_length,delta_film,filmAndDropcell
    open (unit=55,file='input.dat')

    read(55,*) trash
    read(55,*) Pe
    read(55,*) trash
    read(55,*) filmdeposition
    read(55,*) trash
    read(55,*) d_0
    d_0 = d_0*1d-9
    read(55,*) trash
    read(55,*) radius
    if (filmdeposition.eq.1) then
      radius = radius*d_0
    else
      radius = radius*1d-6
    endif

    read(55,*) trash
    read(55,*) trash
    if (filmdeposition.eq.1) then
      read(55,*) trash
      read(55,*) trash
      read(55,*) trash
      read(55,*) delta_film
      cap_length = 0d0
      delta_film = delta_film*d_0
      filmAndDropcell = delta_film
    else
      read(55,*) cap_length
      read(55,*) trash
      read(55,*) trash
      read(55,*) delta_film
      cap_length = cap_length*1d-6
      delta_film = delta_film*1d-6
      filmAndDropcell = delta_film
    endif
    read(55,*) trash
    read(55,*) trash
    read(55,*) geometry
    read(55,*) trash
    read(55,*) trash
    read(55,*) nrmax
    read(55,*) trash
    read(55,*) nphimax
    read(55,*) trash
    read(55,*) nzmax
    read(55,*) trash
    read(55,*) dzfirst
    dzfirst = dzfirst*d_0
    zmax = filmAndDropcell+cap_length
    dz = zmax/dfloat(nzmax)
    skipIncludingCell = nint(filmAndDropcell/dz)
    test = (filmAndDropcell-dfloat(skipIncludingCell)*dz)/filmAndDropcell
    if (abs(test).gt.1d-12) then
      write(*,*) 'error: skipIncludingCell*dz does not equate filmAndDropcell!'
      stop_program = 1
    endif
! **** OLD
!    dzfirst = dzfirst*d_0
!    zmax = delta_film+cap_length
!    dz = zmax/dfloat(nzmax)
!    skipIncludingCell = nint(delta_film/dz)
!    test = delta_film-dfloat(skipIncludingCell)*dz
!    if (abs(test).gt.1d-4) then
!      write(*,*) 'error: skipIncludingCell*dz does not equate delta_film!'
!      stop_program = 1
!    endif

    close(55)
end
```

F.3.11 Number_balance.f90

```
subroutine number_balance()
use variables_deposition
```

```
implicit none
    if (part_pos(1,1).eq.0) then
      balance=0
    else
      balance = inserted-size(part_pos,1)-lost_side-lost_bottom-dropped
    endif
end
```

F.3.12 Variables.f90

```
module variables_deposition

real*8 dr ! grid spacing - cylindrical coordinates
    real*8 dz ! "
    real*8 dphi ! "
    real*8 pi
    real*8 actual_time ! - integrated time
    real*8 radius ! radius of tube
    real*8 zmax ! maximum deposition depth - particles below this depth are reintroduced into the top of the tube

! Particle positions are taken as their center position

    real*8 r0(3) ! Particle 0 (moving particle) position before integration
    real*8 re(3) !  "  after   "
real*8 ri(3) ! Particle i (any particle) position fixed
    real*8 rc(3) ! Particle 0-i collision position

    real*8 d_0 ! Size of particle 0
    real*8 d_i ! Size of particle i
    real*8 d_0pseudo ! for geometry = 0
    real*8 velocity(3)
    real*8 rhoPart
    real*8 muFluid
    real*8 Tgas
    real*8 kB
    real*8 porePlugging0
    real*8 Rgas
    real*8 RhoFluid
    real*8 MwFluid
    real*8 Pressure
    real*8 intTime_nl ! integrated neighbour list time (searching for slowdowns)
    real*8 intTime_int ! integrated integrator time (-"-)

    real*8, dimension(:,:), allocatable :: part_pos(:,:) ! Particle storage matrix - initially 1 row and 4 columns
! The particle storage matrix is made as (cell_number,x,y,z)
                                                          ! where x,y,z is the position in cartesian coordinates of a stored
                                                          ! particle

    real*8, dimension(:,:), allocatable :: neighbour_list(:,:)
    real*8, dimension(:,:), allocatable :: debug_cell(:,:)
    integer*4 nrmax ! grid resolution
    integer*4 nzmax ! "
    integer*4 nphimax ! "
    integer*4 nrcrit  ! crititical grid position at which arc length = 2d_0
    integer*4 dropped
    integer*4 dropped_max
    integer*4 inserted
    integer*4 inserted_max
    integer*4 lost_top
    integer*4 lost_bottom
    integer*4 lost_max
    integer*4 balance
    integer*4 stop_program
    integer*4 debug
    integer*4 geometry
    integer*4 moving
    integer*4 continuation
    integer*4 timeLimitReached
    integer*4 plugging
    integer*4 iCond
    integer*4 skipfirst
    integer*4 firstcelldep
    integer*4 lost_side
    integer*4 lost
    integer*4 filmdeposition
    integer*4 skipIncludingCell
    integer*4 brutecheck
    integer*4 numdif
    integer*4 ncellold
    integer*4 empty_old
    integer*4 iter
    integer*4 test_moves
```

```
    integer*4 difcoef

real*8 lambda ! Free mean path of gas
    real*8 ugasmax
    real*8 Dp ! particle diffusion coefficient
    real*8 dt ! time step
    real*8 Kn ! Knudsen number
    real*8 Cc ! Cunningham Correction Factor
    real*8 disp
    real*8 setTime
    real*8 time0
    real*8 Pe
    real*8 thickness
    real*8 dzfirst
    real*8 Diffusion_Coef
    real*8 rinlet(3)


    character*10 creation_time,creation_date
    character*10 original_creation_time,original_creation_date

end module variables_deposition

subroutine init_variables()
use variables_deposition
    implicit none
    real*8 rcrit

    pi = 3.141592653589793d0
    call read_input()
! ---- Particle size

!    d_0 = 1d-7 ! moving particle     ! use data-file from now on
    d_i = d_0 ! size of particle i (size-distribution to be included in a later vesion)

! ----- Geometry and grid settings

!    radius = 2d-6  ! use data-file from now on
! zmax = 100d-6 ! use data-file from now on
!    nrmax = 4 ! grid resolution ! use data-file from now on
!    nzmax = 10 !  "  ! use data-file from now on
!    nphimax = 4!  "     ! use data-file from now on
!    geometry = 1 ! set to 0 if geometry is turned off- in principle the model should behave as the Hinds model
dr = radius/dfloat(nrmax) ! grid spacing
!    dz = zmax/dfloat(nzmax)  ! "
    dphi = 2d0*pi/nphimax ! "
    rcrit = 2d0*d_0/dphi
    nrcrit = int(rcrit/dr)+1
    write(*,*) 'nrcrit, nrmax: ', nrcrit, nrmax

    skipfirst = 1 ! leave out geometry from inlet cells - define size below
!    skipIncludingCell = 10 ! CHANGED 3/12-2007 - 2 originally
!    filmdeposition = 0
    if ((geometry.eq.0).or.(filmdeposition.eq.1)) then
      skipfirst = 0
      skipIncludingCell = 0
    endif

! ----- Values to define pore-plugging, run-time etc.

    dropped = 0
    inserted = 0
    lost_top = 0
    lost_side = 0
    lost_bottom = 0
    lost = 0
    dropped_max = 100000 ! set these to control run time
    lost_max = 100000000
    inserted_max = 1000
    plugging = 0
    iCond = 0
    porePlugging0 = 1
    time0 = 0d0
    firstcelldep = 0

! ----- Initialize the particle position matrix

    allocate (part_pos(1,4))
    part_pos(1,1) = 0
    part_pos(1,2) = 0
    part_pos(1,3) = 0
    part_pos(1,4) = 0
    allocate (neighbour_list(1,3))
    neighbour_list(1,1) = 0
```

```
    neighbour_list(1,2) = 0
    neighbour_list(1,3) = 0

! ----- Other variables
    difcoef = 0 ! to calculate the diffusion coefficient set difcoef = 1 and geometry = 0
    stop_program = 0
    empty_old = 0
debug = 0 ! Set to 0 always
    brutecheck = 0 ! set to 0 if you are sure there will be no particle-overlapping. Greatly increases calculation speed
    actual_time = 0 ! time at t=t0
    kB = 1.380650524d-23 ! Boltzmann constant (J/K)
    setTime = 7.5d0*60d0 ! time (minutes) allowed to run in total
!    setTime = 1d16 ! CHANGED 3/12-2007
    continuation = 0 ! set to 1 if program should continue from last run
    timeLimitReached = 0
    allocate (debug_cell(1,1))
    intTime_nl = 0d0
    intTime_int = 0d0

! ----- Fluid and particle properies

!    Pe = 0.1d0 ! Peclet Number ! use data-file from now on
    Tgas = 298d0 ! K
    Rgas = 8.3145d0
Pressure = 101325d0 ! pressure
    MwFluid = 28.97d-3 ! molecular weight mol/kg
    rhofluid = Pressure*MwFluid/(Rgas*Tgas)
    muFluid = 0.000001425d0*Tgas**0.5039d0/(1d0+108.3d0/Tgas)
    lambda = muFluid/rhofluid*dsqrt(pi*MwFluid/(2d0*Rgas*Tgas))
    rhoPart = 1000 ! kg/m3
    Kn = 2*lambda/d_0
    Cc = 1+Kn*(1.257+0.4*exp(-1.1/Kn)) ! Cunningham correction factor
    Dp = 1.38066e-23*Tgas/(3*pi*muFluid*(d_0))*Cc
    ugasmax = 2d0*Pe*Dp/(0.5d0*d_0)

! ----- difcoef = 1  write to screen
if (difcoef.eq.1) then
      geometry = 0d0
      ugasmax = 0d0
      write(*,*) 'Calculating diffusion coefficient'
    endif


! ----- Values relating to the time step

    disp = 0.5d0
    dt = (disp*d_0)**2/(2d0*Dp) ! time step

    if (geometry.eq.0) then
      d_0pseudo = d_0
      d_0 = 0
    endif


! ----- Initialization of the r0 vector

    r0(1) = 0
    r0(2) = 0
    r0(3) = 0

! ----- Initilization of the velocity vector
    velocity(1) = 0
    velocity(2) = 0
    velocity(3) = 0

end
```

F.3.13 Functions.f90

```
function cell_number(nr,nphi,nz)
    use variables_deposition
    implicit none
    real*8 cell_number
    integer*4 nr,nz,nphi

    cell_number = (nr-1)*nphimax+(nphi-1)+(nrmax*nphimax)*(nz-1)+1

end

function simple_sign(x)
```

```
implicit none
    integer*4 simple_sign,i
    real*8 x

    i = int(x/abs(x))

    if (i.eq.(-1)) then
      simple_sign = -1
    else
      simple_sign = 1
    endif
end


function grid_size(dr,dphi)
    implicit none
    real*8 grid_size,dr,dphi

    grid_size = dr*sin(dphi)
end

! ******************************************************************************** $PAGE *
! * NAME: D_theoretical
! * full coalescence of particles upon collision
! ******************************************************************************************
real*8 function D_theoretical(x)
use variables_deposition
implicit none
real*8 alpha, friction,x

alpha = 18d0 * muFluid / (rhoPart * Cc * x**2)
friction = alpha * (x**3)/6*pi*rhoPart
D_theoretical = kB * Tgas / friction
end

subroutine crossp(x2,x1,xc) !x2 X x1
implicit none
    real*8 x1(3),x2(3),xc(3)

    xc(1) = x2(2)*x1(3)-x2(3)*x1(2)
    xc(2) = x2(3)*x1(1)-x2(1)*x1(3)
    xc(3) = x2(1)*x1(2)-x2(2)*x1(1)
end

function veclength(x1)
implicit none
    real*8 veclength
    real*8 x1(3)

    veclength = dsqrt(x1(1)**2+x1(2)**2+x1(3)**2)
end
```

F.3.14 Auxillery.f90

```
! ***** Hack to initialize the random seed. Based on the system clock a seed is generated and fed to the
! ***** random number generator, required before call to Random_Number()

SUBROUTINE init_random_seed()
            INTEGER :: i, n, clock
            INTEGER, DIMENSION(:), ALLOCATABLE :: seed
            real*8 timeNowReal,ms
            character*12 timeNowChar,dummy1

            CALL RANDOM_SEED(size = n)
            ALLOCATE(seed(n))

            call date_and_time(dummy1,timeNowChar)
            read(timeNowChar,*) timeNowReal
            ms = 1000d0*(timeNowReal-dfloat(int(timeNowReal))) ! milliseconds included, in case program starts
                ! more or less at the same time (for Gonzales)
            clock = int(timeNowReal) + int(ms)
            seed = clock + 37 * (/ (i - 1, i = 1, n) /)
            CALL RANDOM_SEED(PUT = seed)

            DEALLOCATE(seed)
END SUBROUTINE


! ***** This subroutine calculates 6 random gaussian distributions based on a Box Müller approach.
! ***** The below is basically a copy from M. Heine "Particle Dynamics at High Aerosol Concentrations
! ***** and Production Rates", PhD thesis, ETHZ, 2007.
```

```
subroutine gauss(y)
use variables_deposition
    implicit none
    real*8 q(6)
    real*8 rgauss(6),y(6),temp,temp2,sigma
    integer*4 i

    sigma = 1d0
    do i = 1,6
      call random_number(q(i))
      if (mod(i,2).eq.0) then
        temp = sigma*dsqrt(-2d0*dlog(q(i-1)))
        temp2 = 2d0*pi*q(i)
        rgauss(i-1) = temp*dcos(temp2)
        rgauss(i) = temp*dsin(temp2)
      endif
    enddo
    y = rgauss
end

! ***** This subroutine sorts an array in ascending order
! ***** Program copied off "http://perso.orange.fr/jean-pierre.moreau/Fortran/sort1_f90.txt"
! ***** orignally by Jean-Pierre Moreau. Modified 18-10-2007 by Tobias Dokkedal Elmøe.

SUBROUTINE PIKSRT(N,ARR)
  integer*4 N,j,i
  real*8 ARR(N),a

  do j=2, N
    a=ARR(j)
    do i=j-1,1,-1
      if (ARR(i)<=a) goto 10
      ARR(i+1)=ARR(i)
    end do
i=0
10  ARR(i+1)=a
  end do
  return
END

! ***** This subroutine checks the current time and compares it to the start time.
! ***** This is required for long calculations as a KILL signal will be given from the cluster server
! ***** if the calculations take too long

subroutine check_time(position)
use variables_deposition
    implicit none
    character*10 run_time,run_date
    real*8 x1,x2,difference
    integer*4 position
    integer*4 hour1,minute1
    integer*4 hour2,minute2

    call date_and_time(run_date,run_time)

    read(creation_time,*) x1 ! time is converted to real
    read(run_time,*) x2

    hour1 = int(x1/10000)
    minute1 = int((x1-dfloat(hour1)*10000)/100)
    hour2 = int(x2/10000)
    minute2 = int((x2-dfloat(hour2)*10000)/100)

    if (hour2.lt.hour1) then
      hour2=24d0+hour2
    endif

    difference = dfloat(hour2*60-hour1*60+minute2-minute1)

    call number_balance()
    if (balance.ne.0) then
      position = 20
    else
      position = 10
    endif

    if (difference .ge. setTime) then
      stop_program = 1
      write(*,*) 'Time limit reached. Stop program set to : ', stop_program
      timeLimitReached = 1
      position = 30
    endif
end
```

```
subroutine check_continuation()
    use variables_deposition
    implicit none

    character*50 trash

    read(50,*) trash
    read(50,*) continuation
end

subroutine acc_dist(restart)
use variables_deposition
    implicit none
    integer*4 restart
    real*8 dx,diffusivity,rer0(3),veclength

    if (restart.eq.1) then
      numdif = 0
      return
    else
      write(55,*) Diffusion_Coef,inserted
!      write(*,*) diffusivity,inserted
    endif
end

subroutine diffusivity()
use variables_deposition
    implicit none
    real*8 rvec(3),dx,veclength

    rvec = re-rinlet
    dx = veclength(rvec)
    numdif = numdif+1
    Diffusion_Coef = dx**2/(6d0*dt*numdif)
end

subroutine count_time(startTime,integrator)
use variables_deposition
implicit none
    character*12 startTime,nowTime,dummy1
    real*8 realTime0,realTime,ms1,ms0
    integer*4 integrator,hour0,hour1,minute1,minute0,seconds1,seconds0

    call date_and_time(dummy1,nowTime)

    read(nowTime,*) realTime
    read(startTime,*) realTime0

    hour1 = int(realTime/10000)
    minute1 = int((realTime-dfloat(hour1)*10000)/100)
    seconds1 = int(realTime)-(hour1*10000+minute1*100)
    ms1 = (realTime-dfloat(int(realTime)))

    hour0 = int(realTime0/10000)
    minute0 = int((realTime0-dfloat(hour0)*10000)/100)
    seconds0 = int(realTime0)-(hour0*10000+minute0*100)
    ms0 = (realTime0-dfloat(int(realTime0)))

    realTime = dfloat(hour1)*3600d0+dfloat(minute1)*60d0+dfloat(seconds1)+ms1
    realTime0 = dfloat(hour0)*3600d0+dfloat(minute0)*60d0+dfloat(seconds0)+ms0

    if (integrator.eq.0) then
      intTime_nl = (realTime-realTime0)+intTime_nl
    else
      intTime_int = (realTime-realTime0)+intTime_int
    endif

end
```

Appendix G

G.1 Program structure

Figure G.1 shows a schematic of the program structure. The boxes indicate subroutines. A particle is inserted initially. After moving, the position is checked. Depending on the position of the particle, two different routes are followed in the program.

Particle in capillary domain

In the case the particle is in the capillary domain, particle-particle collision is checked first. If there is no possibility of collision, then particle-wall collision is checked. If no collision is possible there, then the particle is moved again. If collision can occur with another particle (or several), the corresponding set of collision coordinates are calculated. Furthermore, the possibility of simultaneous collision with the wall is checked and the right collision coordinates are selected based on the criterion shown in eq. 5.17. The particle is stored in the particle matrix (`part_pos`), which contains the position of all deposited particles sorted by their cell number (for use in the neighbour list system).

G.1.1 Particle in the cake domain

If the particle is located in the cake domain, it is not possible for the particle to "stick" to the wall. As for the cake-domain, particle-particle collision is checked first. If no collision is possible, then it is checked whether or not the crosses the boundary. If so, then the particle is reintroduced using symmetrical boundary conditions.

G.1.2 Program initilization

Specifications of the particle size, capillary radius, Pe number, etc. are given in the file `input.dat`. The program has a default time-limit of 7.5 hours (set in `variables.f90`), to accommodate the requirements of the Gonzales cluster (ETH Zürich), which was used to run the program. The program

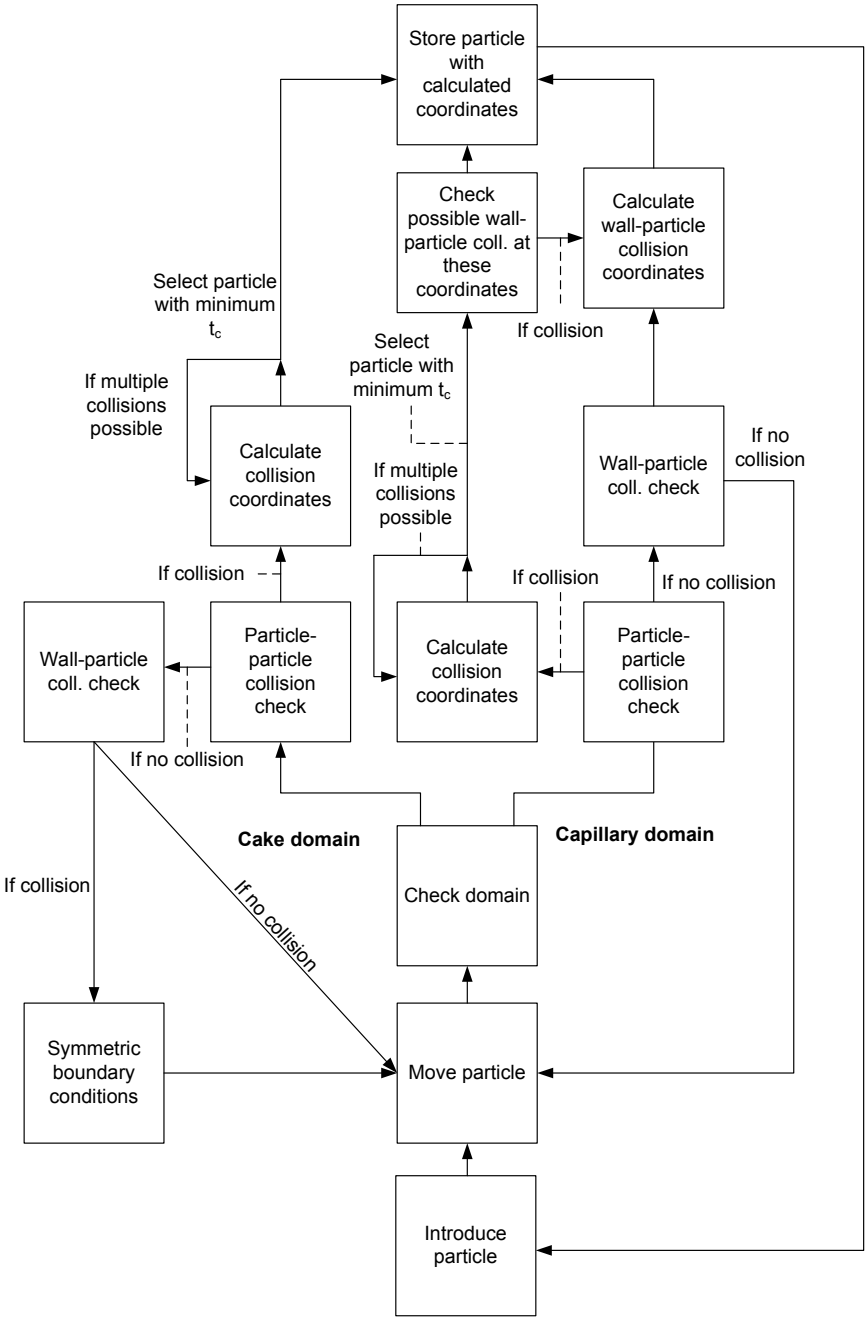


Figure G.1 Schematic of the program structure. The subroutines are represented by blocks and the lines inbetween them represent the "if" checks, unless otherwise stated on the figure.

stores all variables, such as position of the (if) moving particle along with its velocity vector, random seed values, etc. in the `temp.dat` file. The particle position matrix is stored in a separate file, with the coordinates saved in double precision (16 digits). By resubmitting the program, the calculations are continued automatically. If the program finishes the calculations within the time-limit (i.e. if the cake grows to δ_{film}), the output files are saved.