Cement Paste Degradation Modeling with Cem-GEMS: Complete Project Guide

Project: Multi-environment reaction-transport modeling of fly ash blended cement paste under salt attack
Tool: Cem-GEMS (GEMS-PSI) with Python interface (xGEMS)
Duration: 60 days erosion simulation at 20°C
Date: February 2026

Project Overview

Objective

Model the evolution of hydration products and pore solution chemistry in Portland cement + fly ash paste exposed to:

1. **Pure water** (immersion only / with applied pressure)
2. **70 g/L NaCl solution** (immersion only / with applied pressure)
3. **10 g/L Na_2SO_4 + 70 g/L NaCl mixed solution** (immersion only / with applied pressure)

Total scenarios: 6 (3 solutions × 2 hydraulic conditions)

Key Outputs Required

- Time evolution (0-60 days) of phase contents: portlandite, C-S-H gel, ettringite, AFm phases, Friedel's salt, gypsum, M-S-H, unhydrated clinker
- Pore solution composition: pH, [Ca^{2+}], [Na^+], [Cl^-], [SO_4^{2-}], [OH^-], [AlO_2^-], [SiO_3^{2-}]
- Comparison with literature degradation stages (4-stage leaching model, sulfate attack mechanisms)
- Complete modeling report with parameters, assumptions, and validation against XRD/TGA data

Software Requirements

Core Software

- **GEMS-PSI / Cem-GEMS** (web interface or desktop version)
  - Download: https://cemgems.org or https://gems.web.psi.ch
  - Version: GEMS 3.3 or later with CEMDATA18 database
- **Python 3.8+** with packages:
  - `xgems` (Python interface to GEMS ChemicalEngine)
  - `numpy`, `pandas`, `matplotlib`, `scipy`
  - Install: `pip install xgems numpy pandas matplotlib scipy`

Thermodynamic Databases

- **Nagra/PSI TDB** (base aqueous species and minerals)
- **CEMDATA18** (cement hydration phases, C-S-H, AFm/AFt solid solutions)
- Both are bundled with GEMS/Cem-GEMS installation

Reference Documents

Keep these papers accessible throughout the project:

1. Jacques et al. (2010) - Leaching model with rain/soil water
2. Li et al. (2020) - Sulfate attack under electric fields
3. Zuo et al. (2023) - Combined NaCl + $MgSO_4$ attack on alkali-activated slag
4. Lothenbach et al. (2019) - CEMDATA18 database documentation

Phase 1: Thermodynamic Basis Setup

What You Need to Do

1. Install GEMS/Cem-GEMS and verify database access
2. Configure project with CEMDATA18 + Nagra/PSI databases
3. Set temperature to 20°C and standard pressure (1 bar)
4. Verify C-S-H model (CSHQ or CSH_ss) and AFm/AFt solid solutions are enabled

Files Created

File: project_config.json (or GEMS project file .gar)
Location: ./gems_project/

Content:
```
{
"project_name": "CementFlyAsh_SaltAttack",
"databases": ["CEMDATA18", "NagraPSI"],
"temperature_C": 20,
"pressure_bar": 1,
"csh_model": "CSHQ",
"solid_solutions": ["AFm_ss", "AFt_ss", "CSH_ss"],
"suppressed_phases": ["C3AS0.8H4.4", "thaumasite"]
}
```

Inputs

- CEMDATA18 database files (.dat, .dch)
- Nagra/PSI database files
- GEMS configuration preferences

Outputs

- Initialized GEMS project with thermodynamic data loaded
- Log file confirming database consistency at 20°C

Commands / Actions

In Cem-GEMS web interface:

1. Create new project → Select "Concrete/Cement" template
2. Settings → Database → Add CEMDATA18 and NagraPSI
3. Settings → Defaults → Set T=20°C, suppress thaumasite and $C_3AS_{0.8}H_{4.4}$

In Python (xGEMS):
```
from xgems import ChemicalEngine
engine = ChemicalEngine(database='cemdata18')
engine.set_temperature(293.15) # 20°C in Kelvin
```

Success Criteria

- [ ] GEMS project loads without errors
- [ ] Phase list includes: portlandite, C-S-H, ettringite, monosulfate, Friedel's salt, hydrotalcite
- [ ] Temperature-corrected equilibrium constants visible at 20°C

Phase 2: Material Recipe Definition

What You Need to Do

1. Define cement clinker composition from oxide and mineral data
2. Define fly ash composition (glass phase + crystalline phases)
3. Create paste recipe with water/binder ratio
4. Set up 28-day hydration baseline equilibrium

Files Created

File 1: `cement_composition.json`

Location: `./materials/`

Content (based on PI 52.5 from Xiang-Guan-Can-Shu-1.docx):

```
{
"cement_type": "PI_52.5",
"oxides_wt_percent": {
"CaO": 63.63,
"SiO2": 17.37,
"Al2O3": 4.20,
"Fe2O3": 3.89,
"MgO": 2.15,
"SO3": 2.05,
"Na2O": 0.34,
"K2O": 0.38,
"TiO2": 0.26
},
"minerals_wt_percent": {
"C3S": 61.8,
"C2S": 8.6,
"C3A": 1.7,
"C4AF": 16.2,
"K2SO4": 0.2,
"bassanite": 3.3,
"CaCO3": 0.5
},
"density_g_cm3": 3.15
}
```

File 2: `flyash_composition.json`

Location: `./materials/`

Content:
```
{
"flyash_type": "Class_F",
"oxides_wt_percent": {
"Al2O3": 41.56,
"SiO2": 43.45,
"Fe2O3": 4.93,
"CaO": 4.16,
"MgO": 0.42,
"K2O": 0.67,
"Na2O": 0.012,
"SO3": 0.78,
"TiO2": 1.85
},
"phases_wt_percent": {
"glass": 73.0,
"mullite": 15.0,
"quartz": 12.0
},
"density_g_cm3": 2.35,
"glass_reactivity_note": "Calibrate to match 28d XRD/TGA"
}
```

File 3: `paste_recipe.json`

Location: `./recipes/`

Content:
```
{
"recipe_name": "OPC_FA_Paste",
```

"w_b_ratio": 0.5,
"fly_ash_replacement_percent": 30,
"masses_per_1000cm3": {
"cement_g": 980,
"fly_ash_g": 420,
"water_g": 700
},
"initial_conditions": {
"closed_system": true,
"hydration_time_days": 28,
"temperature_C": 20
}
}

File 4: hydration_28d.py

Location: ./scripts/

Purpose: Run 28-day hydration to establish initial state

Content:
import json
import numpy as np
from xgems import ChemicalEngine

Load material data

with open('../materials/cement_composition.json') as f:
cement = json.load(f)
with open('../materials/flyash_composition.json') as f:
flyash = json.load(f)
with open('../recipes/paste_recipe.json') as f:
recipe = json.load(f)

Initialize GEMS engine

engine = ChemicalEngine(database='cemdata18')
engine.set_temperature(293.15)

Define system composition (simplified for 1 kg paste)

```
system_composition = {
# Cement oxides scaled by mass
'CaO': cement['oxides_wt_percent']['CaO'] *
recipe['masses_per_1000cm3']['cement_g'] / 100,
'SiO2': (cement['oxides_wt_percent']['SiO2'] *
recipe['masses_per_1000cm3']['cement_g'] +
flyash['oxides_wt_percent']['SiO2'] * recipe['masses_per_1000cm3']
['fly_ash_g']) / 100,
# ... add all oxides
'H2O': recipe['masses_per_1000cm3']['water_g']
}
```

Set hydration degrees (initial guess, will calibrate)

```
hydration_degrees = {
'C3S': 0.95,
'C2S': 0.65,
'C3A': 1.0,
'C4AF': 0.70,
'FA_glass': 0.20 # Fly ash glass reactivity at 28d
}
```

Calculate equilibrium at 28d

```
result = engine.equilibrate(system_composition, suppress=
['C3AS0.8H4.4', 'thaumasite'])
```

Save baseline state

```
baseline = {
'phases': result['phases'],
'pore_solution': result['aqueous'],
'pH': result['pH'],
'porosity': result['porosity']
}

with open('../outputs/baseline_28d.json', 'w') as f:
json.dump(baseline, f, indent=2)

print("28-day hydration complete. Baseline saved.")
```

Inputs

- Cement oxide and mineral composition from XRD/XRF
- Fly ash oxide composition and phase fractions (glass, mullite, quartz)
- Water/binder ratio (0.5 assumed, adjust if specified)
- Fly ash replacement level (30% assumed, adjust as needed)

Outputs

- `baseline_28d.json`: Phase assemblage and pore solution at 28 days before exposure
- `baseline_phases.csv`: Tabular phase data for validation against XRD
- `baseline_poresolution.csv`: Aqueous species concentrations

Validation

Compare `baseline_phases.csv` with client's 28-day XRD/TGA:

- Portlandite content (from TGA ~105-450°C mass loss)
- AFm/AFt phases (XRD peak intensities)
- Unhydrated clinker (residual C_3S, C_2S peaks)

Adjust hydration degrees in `hydration_28d.py` until model matches measurements within ±10%.

Success Criteria

- [ ] Baseline portlandite within 10% of TGA measurement
- [ ] Ettringite/monosulfate ratio matches XRD qualitative assessment
- [ ] pH in pore solution ~13.5-13.8 (typical for OPC paste)
- [ ] Unhydrated clinker consistent with expected 28-day hydration

---

Phase 3: External Solutions and Boundary Conditions

What You Need to Do

1. Define three external solution compositions
2. Set up "immersion" vs "pressure" renewal rate parameters
3. Decide stepping scheme (time-based or cumulative water)

Files Created

File 1: `external_solutions.json`

Location: `./solutions/`

Content:
```
{
"pure_water": {
"description": "Deionized water, pH neutral",
"composition_mol_L": {
"H2O": 55.5,
"dissolved_CO2_atm": 3.5e-4
},
"pH_initial": 7.0
},
"NaCl_solution": {
"description": "70 g/L NaCl (1.2 mol/L)",
"composition_mol_L": {
"Na+": 1.2,
"Cl-": 1.2,
"H2O": 55.5
},
"pH_initial": 7.0
},
"mixed_salt_solution": {
"description": "10 g/L Na2SO4 + 70 g/L NaCl",
"composition_mol_L": {
"Na+": 1.34,
"Cl-": 1.2,
"SO4_2-": 0.07,
"H2O": 55.5
},
"pH_initial": 7.0
}
}
```

File 2: `process_parameters.json`

Location: `./process_config/`

Content:

```
{
"simulation_duration_days": 60,
"temperature_C": 20,
"immersion_conditions": {
"description": "Static soaking with slow pore solution renewal",
"external_water_per_step_kg": 0.5,
"step_interval_days": 3,
"total_steps": 20
},
"pressure_conditions": {
"description": "Enhanced flow with faster pore solution renewal",
"external_water_per_step_kg": 2.0,
"step_interval_days": 3,
"total_steps": 20,
"note": "Higher water/solid ratio simulates increased hydraulic
pressure"
},
"leaching_model": {
"type": "mass_balance",
"description": "Equilibrate paste with external solution, then replace
```

solution while retaining solid phases",
"reference": "Jacques et al. 2010 Cement Concrete Res"
}
}

Inputs

- Solution concentrations from experimental design (Xiang-Guan-Can-Shu-1.docx)
- Literature guidance on leaching rates (Jacques et al. cumulative water values)
- Client description of "pressure" effect (faster pore solution renewal)

Outputs

- Validated solution recipes for GEMS input
- Renewal rate parameters that differentiate immersion vs pressure

Key Decisions

Stepping variable choice:

- **Option A (recommended)**: Cumulative external water per 1000 cm^3 paste
  - Maps directly to Jacques et al. methodology
  - Pressure = 4× water per step vs immersion
  - Convert cumulative water to equivalent time for 60-day target
- **Option B**: Fixed time steps with concentration boundary
  - Simpler but less accurate for comparing pressure effect
  - Use if xGEMS doesn't support custom lead variables

Document chosen approach in `process_parameters.json` and report.

Success Criteria

- [ ] Solution compositions convert to valid GEMS input (charge balanced)
- [ ] Immersion vs pressure parameter differs by factor of 2-4×
- [ ] Total simulation reaches ~60 days equivalent exposure

Phase 4: Process Implementation (6 Scenarios)

What You Need to Do

For each of 6 scenarios, create a Python script that:

1. Loads 28-day baseline state
2. Iteratively equilibrates with external solution
3. Replaces solution between steps
4. Records phase assemblage and pore solution at each step
5. Saves time-series output

Files Created

Create 6 scripts in `./scripts/` directory:

1. `run_PW_immersion.py`
2. `run_PW_pressure.py`
3. `run_NaCl_immersion.py`
4. `run_NaCl_pressure.py`
5. `run_mixed_immersion.py`
6. `run_mixed_pressure.py`

Template Script: `run_PW_immersion.py`

Location: `./scripts/`

Content:

```
"""
Pure Water Leaching - Immersion Condition
Simulates 60-day exposure with low renewal rate
"""

import json
import numpy as np
import pandas as pd
from xgems import ChemicalEngine
```

================ CONFIGURATION ================

```
SCENARIO_NAME = "PW_Immersion"
SOLUTION_TYPE = "pure_water"
CONDITION_TYPE = "immersion_conditions"
```

Load configurations

```
with open('../solutions/external_solutions.json') as f:
solutions = json.load(f)
with open('../process_config/process_parameters.json') as f:
params = json.load(f)
with open('../outputs/baseline_28d.json') as f:
baseline = json.load(f)
```

Extract parameters

```
solution = solutions[SOLUTION_TYPE]
process = params[CONDITION_TYPE]
n_steps = process['total_steps']
water_per_step = process['external_water_per_step_kg']
step_interval = process['step_interval_days']
```

Initialize GEMS engine

```
engine = ChemicalEngine(database='cemdata18')
engine.set_temperature(293.15)
```

================ INITIALIZE STATE ================

current_solid = baseline['phases'].copy()
cumulative_water = 0.0
time_days = 0.0

Storage for results

results = {
'time_days': [],
'cumulative_water_kg': [],
'phases': [],
'pore_solution': [],
'pH': []
}

================ SIMULATION LOOP ================

print(f"Starting {SCENARIO_NAME} simulation...")
print(f"Steps: {n_steps}, Water per step: {water_per_step} kg\n")

for step in range(n_steps + 1):
print(f"Step {step}/{n_steps} - Time: {time_days:.1f} d - Cumulative water: {cumulative_water:.2f} kg")

```
# ===== Equilibrate current solid with external solution =====
system = current_solid.copy()
system.update({
    'H2O': water_per_step * 1000,  # Convert kg to g
    'Na+': solution['composition_mol_L'].get('Na+', 0) * water_per_step,
    'Cl-': solution['composition_mol_L'].get('Cl-', 0) * water_per_step,
    'SO4_2-': solution['composition_mol_L'].get('SO4_2-', 0) * water_per_step,
})
```

```
# Run equilibrium calculation
result = engine.equilibrate(system, suppress=['C3AS0.8H4.4', 'thaumasite'])

# ===== Record state =====
results['time_days'].append(time_days)
results['cumulative_water_kg'].append(cumulative_water)
results['phases'].append(result['phases'].copy())
results['pore_solution'].append(result['aqueous'].copy())
results['pH'].append(result['pH'])

# Print key phases
print(f"  Portlandite: {result['phases'].get('portlandite', 0):.3f} mol")
print(f"  Ettringite: {result['phases'].get('ettringite', 0):.3f} mol")
print(f"  C-S-H: {result['phases'].get('CSH', 0):.3f} mol")
print(f"  pH: {result['pH']:.2f}\n")

# ===== Update solid composition for next step =====
# Keep solid phases, discard aqueous phase (simulate solution replacement)
current_solid = {k: v for k, v in result['phases'].items()
                 if k not in ['H2O', 'aqueous']}

# Update counters
cumulative_water += water_per_step
time_days += step_interval
```

================ SAVE RESULTS ================

output_dir = f'../outputs/{SCENARIO_NAME}/'
import os
os.makedirs(output_dir, exist_ok=True)

Save raw data

```
with open(f'{output_dir}raw_results.json', 'w') as f:
json.dump(results, f, indent=2)
```

Convert to DataFrame for easier analysis

```
phase_df = pd.DataFrame([
{
'time_days': results['time_days'][i],
'cumulative_water_kg': results['cumulative_water_kg'][i],
'portlandite': results['phases'][i].get('portlandite', 0),
'CSH': results['phases'][i].get('CSH', 0),
'ettringite': results['phases'][i].get('ettringite', 0),
'monosulfate': results['phases'][i].get('monosulfate', 0),
'friedel': results['phases'][i].get('friedel', 0),
'gypsum': results['phases'][i].get('gypsum', 0),
'calcite': results['phases'][i].get('calcite', 0),
'C3S': results['phases'][i].get('C3S', 0),
'C2S': results['phases'][i].get('C2S', 0),
'pH': results['pH'][i]
}
for i in range(len(results['time_days']))
])
phase_df.to_csv(f'{output_dir}phase_evolution.csv', index=False)
```

```
pore_df = pd.DataFrame([
{
'time_days': results['time_days'][i],
'pH': results['pH'][i],
'Ca_mol_L': results['pore_solution'][i].get('Ca+2', 0),
'Na_mol_L': results['pore_solution'][i].get('Na+', 0),
'K_mol_L': results['pore_solution'][i].get('K+', 0),
'Cl_mol_L': results['pore_solution'][i].get('Cl-', 0),
'SO4_mol_L': results['pore_solution'][i].get('SO4-2', 0),
'OH_mol_L': results['pore_solution'][i].get('OH-', 0),
'AlO2_mol_L': results['pore_solution'][i].get('AlO2-', 0),
'SiO3_mol_L': results['pore_solution'][i].get('SiO3-2', 0)
}
for i in range(len(results['time_days']))
])
pore_df.to_csv(f'{output_dir}pore_solution_evolution.csv',
index=False)

print(f"\n{SCENARIO_NAME} simulation complete!")
print(f"Results saved to {output_dir}")
```

Replicate for 5 other scenarios, changing:

- `SCENARIO_NAME`
- `SOLUTION_TYPE` (pure_water, NaCl_solution, mixed_salt_solution)
- `CONDITION_TYPE` (immersion_conditions, pressure_conditions)

Inputs

For each script:

- `baseline_28d.json` (initial state)
- `external_solutions.json` (solution composition)
- `process_parameters.json` (renewal rate)

Outputs

For each scenario, in `./outputs/{SCENARIO_NAME}/`:

1. **`raw_results.json`**
   Complete time-series data (all phases, all aqueous species)
2. **`phase_evolution.csv`**
   Key phase contents vs time:
   time_days, cumulative_water_kg, portlandite, CSH, ettringite, monosulfate, friedel, gypsum, calcite, C3S, C2S, pH
3. **`pore_solution_evolution.csv`**
   Pore solution chemistry vs time:
   time_days, pH, Ca_mol_L, Na_mol_L, K_mol_L, Cl_mol_L, SO4_mol_L, OH_mol_L, AlO2_mol_L, SiO3_mol_L

Execution

Run all 6 scenarios:

```
cd scripts/
python run_PW_immersion.py
python run_PW_pressure.py
python run_NaCl_immersion.py
python run_NaCl_pressure.py
python run_mixed_immersion.py
python run_mixed_pressure.py
```

Estimated runtime: 5-30 minutes per scenario (depends on convergence)

Success Criteria

- [ ] All 6 scripts run to completion without errors
- [ ] Portlandite decreases monotonically in all scenarios
- [ ] Ettringite forms in sulfate-containing solutions
- [ ] Friedel's salt forms in NaCl solutions
- [ ] Pressure scenarios show faster phase changes than immersion
- [ ] pH drops from ~13.8 to 10-12 range by 60 days (leaching cases)

Phase 5: Calibration and Validation

What You Need to Do

1. Compare baseline (28d) with client XRD/TGA data
2. Adjust hydration degrees and fly ash reactivity if needed
3. Check that degradation trends match literature stages
4. Perform sensitivity analysis on key parameters

Files Created

File 1: calibration_log.md

Location: ./validation/

Content:

Calibration Log

Baseline 28-day State

Iteration 1 (Initial Guess)

- C_3S hydration: 95%, C_2S: 65%, C_3A: 100%, C_4AF: 70%
- Fly ash glass: 20% reacted
- **Results**:
  - Portlandite: 0.45 mol/1000cm^3 (XRD target: 0.50 ± 0.05)
  - Ettringite: 0.06 mol (XRD: moderate peaks)
  - pH: 13.7 (expected)
- **Action**: Increase C_3S to 97%, FA glass to 25%

Iteration 2 (Adjusted)

- C_3S hydration: 97%, C_2S: 65%, C_3A: 100%, C_4AF: 70%
- Fly ash glass: 25% reacted
- **Results**:
  - Portlandite: 0.48 mol ✓ (within target)
  - Ettringite: 0.07 mol ✓
  - pH: 13.8 ✓
- **Status**: ACCEPTED

Degradation Validation

Pure Water Leaching

- [ ] Stage 1 (Na/K leach): ~0.5 kg cumulative water ✓
- [ ] Stage 2 (Portlandite dissolution): ends at ~70 kg water ✓
- [ ] Stage 3 (C-S-H/AFm/AFt complex): 70-1200 kg water ✓
- [ ] pH drops from 13.8 → 12.5 → 10.5 ✓

NaCl Attack

- [ ] Friedel's salt forms within 7 days ✓
- [ ] AFm → Friedel conversion observed ✓
- [ ] Chloride binding capacity ~10 mg Cl/g paste ✓

Mixed Salt Attack

- [ ] Ettringite + Friedel co-exist initially ✓
- [ ] Gypsum forms at late stage ✓
- [ ] M-S-H detected if Mg present ✓

File 2: `sensitivity_analysis.py`

Location: `./scripts/`

Content:
"""
Sensitivity analysis on key parameters
Vary: fly ash content, water/step, initial portlandite
"""

import numpy as np
import matplotlib.pyplot as plt
from run_PW_immersion import run_scenario # Modularize main script

Define parameter ranges

fa_contents = [20, 30, 40] # % replacement
water_per_step_values = [0.25, 0.5, 1.0, 2.0] # kg
initial_CH = [0.4, 0.5, 0.6] # mol portlandite

Run parametric sweep

results_matrix = []

for fa in fa_contents:
for water in water_per_step_values:
result = run_scenario(

```
fa_replacement=fa,
water_per_step=water,
scenario_name=f'FA{fa}_W{water}'
)
results_matrix.append({
'fa': fa,
'water': water,
'portlandite_loss_rate': result['CH_loss_rate'],
'ettringite_max': result['ettringite_max'],
'pH_final': result['pH'][-1]
})
```

Save and plot

```
import pandas as pd
df = pd.DataFrame(results_matrix)
df.to_csv('../validation/sensitivity_results.csv', index=False)
```

Plot: Portlandite loss rate vs water/step for different FA

```
fig, ax = plt.subplots()
for fa in fa_contents:
subset = df[df['fa'] == fa]
ax.plot(subset['water'], subset['portlandite_loss_rate'],
marker='o', label=f'FA {fa}%')
ax.set_xlabel('Water per step (kg)')
ax.set_ylabel('Portlandite loss rate (mol/day)')
ax.legend()
ax.grid(True)
plt.savefig('../validation/sensitivity_CH_loss.png', dpi=300)
print("Sensitivity analysis complete.")
```

Inputs

- Client XRD/TGA data (portlandite, ettringite, AFm quantification)
- Literature degradation stage definitions (Jacques et al., Li et al.)
- Parameter ranges for sensitivity (fly ash 20-40%, water/step 0.25-2.0 kg)

Outputs

- `calibration_log.md`: Documented iterations and final parameters
- `validation_plots/`: Comparison plots (model vs XRD)
- `sensitivity_results.csv`: Parametric sweep results
- `sensitivity_CH_loss.png`: Example sensitivity plot

Validation Metrics

Baseline (28d):

- Portlandite: model within ±10% of TGA
- Ettringite/monosulfate ratio: qualitative match to XRD
- Unhydrated clinker: ±15% of Rietveld refinement

Degradation trends:

- Stage durations: model within factor of 2 vs Jacques et al.
- Friedel formation: within 7-14 days under NaCl
- Ettringite peak: 7-21 days under sulfate attack

Success Criteria

- [ ] Baseline validation passed for all key phases
- [ ] Degradation follows 4-stage leaching pattern
- [ ] NaCl forms Friedel as dominant chloride phase
- [ ] Mixed salt shows coupling effects (LDH, ettringite + Friedel)
- [ ] Sensitivity analysis identifies critical parameters (FA content, renewal rate)

Phase 6: Data Analysis and Visualization

What You Need to Do

1. Aggregate results from 6 scenarios
2. Create comparison plots (phase evolution, pore solution)
3. Calculate derived metrics (chloride binding, decalcification rate)
4. Generate figures for final report

Files Created

File 1: `plot_phase_evolution.py`

Location: `./scripts/`

Content:

```
"""
Generate phase evolution plots for all scenarios
Compare immersion vs pressure, different solutions
"""

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

scenarios = [
'PW_Immersion', 'PW_Pressure',
'NaCl_Immersion', 'NaCl_Pressure',
'Mixed_Immersion', 'Mixed_Pressure'
]
```

Load data

```
data = {}
for scenario in scenarios:
data[scenario] =
pd.read_csv(f'../outputs/{scenario}/phase_evolution.csv')
```

=========== PLOT 1: Portlandite Evolution ===========

```
fig, axes = plt.subplots(1, 3, figsize=(15, 4))
fig.suptitle('Portlandite Evolution Across Scenarios', fontsize=14,
fontweight='bold')

for i, solution in enumerate(['PW', 'NaCl', 'Mixed']):
ax = axes[i]
for condition in ['Immersion', 'Pressure']:
scenario = f'{solution}{condition}'
df = data[scenario]
ax.plot(df['time_days'], df['portlandite'],
marker='o', label=condition, linewidth=2)
ax.set_xlabel('Time (days)')
ax.set_ylabel('Portlandite (mol)')
ax.set_title(f'{solution.replace("", " ")} Solution')
ax.legend()
ax.grid(True, alpha=0.3)

plt.tight_layout()
plt.savefig('../outputs/plots/portlandite_comparison.png', dpi=300)
plt.close()
```

=========== PLOT 2: Sulfate-Bearing Phases ===========

```
fig, axes = plt.subplots(2, 2, figsize=(12, 10))
fig.suptitle('Sulfate Attack: Ettringite and Gypsum Formation',
fontsize=14, fontweight='bold')

sulfate_scenarios = ['NaCl_Immersion', 'NaCl_Pressure',
'Mixed_Immersion', 'Mixed_Pressure']

for i, scenario in enumerate(sulfate_scenarios):
ax = axes.flat[i]
df = data[scenario]
```

ax.plot(df['time_days'], df['ettringite'],
marker='s', label='Ettringite', linewidth=2)
ax.plot(df['time_days'], df['gypsum'],
marker='^', label='Gypsum', linewidth=2)
ax.plot(df['time_days'], df['monosulfate'],
marker='d', label='Monosulfate', linewidth=2)
ax.set_xlabel('Time (days)')
ax.set_ylabel('Phase content (mol)')
ax.set_title(scenario.replace('_', ' '))
ax.legend()
ax.grid(True, alpha=0.3)

plt.tight_layout()
plt.savefig('../outputs/plots/sulfate_phases.png', dpi=300)
plt.close()

========== PLOT 3: Chloride Binding ==========

fig, ax = plt.subplots(figsize=(8, 6))

for scenario in ['NaCl_Immersion', 'NaCl_Pressure',
'Mixed_Immersion', 'Mixed_Pressure']:
df = data[scenario]
df_pore =
pd.read_csv(f'../outputs/{scenario}/pore_solution_evolution.csv')

```
# Calculate bound chloride (assuming total Cl known)
total_Cl = 1.2 * 0.7  # mol from 70g/L NaCl in 700g water
bound_Cl = total_Cl - df_pore['Cl_mol_L'] * 0.7  # approximate

ax.plot(df['time_days'], bound_Cl,
        marker='o', label=scenario.replace('_', ' '), linewidth=2)
```

ax.set_xlabel('Time (days)')
ax.set_ylabel('Bound Chloride (mol)')
ax.set_title('Chloride Binding Capacity')

```
ax.legend()
ax.grid(True, alpha=0.3)
plt.savefig('../outputs/plots/chloride_binding.png', dpi=300)
plt.close()

print("All plots generated successfully!")
print("Saved to: ../outputs/plots/")
```

File 2: `plot_pore_solution.py`

Location: `./scripts/`

Purpose: Plot pH and major ion evolution

Key plots:

- pH vs time for all 6 scenarios
- [Ca^{2+}] vs time (decalcification indicator)
- [Cl^-] and [SO_4^{2-}] in pore solution (binding assessment)

File 3: `calculate_metrics.py`

Location: `./scripts/`

Purpose: Compute derived quantities

Outputs:

- Decalcification rate (mol Ca lost per day)
- Chloride binding capacity (mg Cl per g paste)
- Cumulative water to reach degradation stages
- Phase volume changes (expansion/shrinkage)

Outputs

In `./outputs/plots/`:

1. `portlandite_comparison.png` - CH evolution across scenarios
2. `sulfate_phases.png` - Ettringite, gypsum, monosulfate
3. `chloride_binding.png` - Bound Cl vs time
4. `pH_evolution.png` - pH trajectories
5. `calcium_leaching.png` - [Ca^{2+}] loss rates
6. `csh_decalcification.png` - C-S-H Ca/Si ratio change

In ./outputs/:

7. summary_metrics.csv - Tabulated degradation metrics
8. stage_boundaries.json - Cumulative water at stage transitions

Execution

```
cd scripts/
python plot_phase_evolution.py
python plot_pore_solution.py
python calculate_metrics.py
```

Success Criteria

- [ ] All plots render without errors
- [ ] Clear differences visible between immersion vs pressure
- [ ] Trends consistent with literature (portlandite → ettringite → gypsum)
- [ ] Chloride binding shows saturation behavior
- [ ] pH drops correlate with portlandite depletion

Phase 7: Final Report Generation

What You Need to Do

1. Write comprehensive modeling report
2. Include all methodology, parameters, results, discussion
3. Compare with literature degradation mechanisms
4. Document assumptions and limitations

Files Created

File 1: FINAL_REPORT.pdf

Location: ./deliverables/

Structure (30-50 pages):

1. Executive Summary (1-2 pages)

- Project scope, 6 scenarios, key findings
- Main phase evolution trends (portlandite loss, ettringite, Friedel)

- Pressure effect quantification (2-4× faster degradation)

2. Introduction (2-3 pages)

- Background on cement durability, salt attack mechanisms
- Fly ash role in hydration and degradation
- Project objectives and deliverables

3. Thermodynamic Modeling Framework (4-5 pages)

- GEMS-PSI software and databases (CEMDATA18, Nagra/PSI)
- C-S-H model (CSHQ), AFm/AFt solid solutions
- Temperature correction (20°C vs standard 25°C)
- Gibbs energy minimization principle
- Suppressed phases (thaumasite, $C_3AS_{0.8}H_{4.4}$) and justification

4. Material Characterization (3-4 pages)

- Cement composition (Table: oxides, minerals)
- Fly ash composition (Table: oxides, glass/mullite/quartz)
- Paste recipe (w/b ratio, FA replacement %)
- 28-day baseline hydration:
  - XRD/TGA validation (plots: measured vs modeled)
  - Phase assemblage (portlandite, C-S-H, AFm, AFt, unhydrated)
  - Pore solution chemistry (pH, $[Ca^{2+}]$, $[Na^+]$, $[K^+]$)

5. Degradation Process Modeling (5-6 pages)

- Mass-balance leaching approach (reference: Jacques et al.)
- External solution compositions (Table: pure water, NaCl, mixed)
- Immersion vs pressure conditions:
  - Renewal rate parameters (water/step, step interval)
  - Physical interpretation (hydraulic pressure → faster exchange)
  - Mapping to 60-day exposure
- Stepping scheme (cumulative water or time-based)
- Numerical implementation (loop structure, convergence criteria)

6. Results: Phase Evolution (8-10 pages)

6.1 Pure Water Leaching

- Figures: Portlandite, C-S-H, calcite vs time
- Four-stage degradation pattern:
    - Stage 1: Alkali leach (Na, K)
    - Stage 2: Portlandite dissolution (pH 13.8 → 12.5)
    - Stage 3: C-S-H/AFm/AFt dissolution (pH 12.5 → 10.5)
    - Stage 4: Calcite buffering
- Cumulative water at stage transitions
- Pressure effect: 3× faster portlandite depletion

6.2 NaCl Attack

- Figures: Friedel, AFm, C-S-H, chloride binding
- Friedel's salt formation kinetics (7-14 days)
- AFm → Friedel conversion (monosulfate depletion)
- Chloride binding capacity (~8-12 mg Cl/g paste)
- Pore solution [Cl^-] evolution (free vs bound)
- Pressure effect: faster Friedel saturation

6.3 Mixed NaCl + Na_2SO_4 Attack

- Figures: Ettringite, Friedel, gypsum, monosulfate
- Coupled sulfate-chloride interactions:
    - Ettringite + Friedel co-precipitation
    - Competition for AFm phases
    - Late-stage gypsum formation
- Comparison with single-salt attacks
- Hydrotalcite/LDH role (if present)
- Enhanced decalcification under combined attack

7. Results: Pore Solution Chemistry (4-5 pages)

- Figures: pH, [Ca^{2+}], [Na^+], [Cl^-], [SO_4^{2-}] vs time
- Decalcification rates (mol Ca leached per day)
- Ionic strength evolution
- Chloride/sulfate partitioning (aqueous vs solid)
- Comparison across scenarios

8. Discussion (5-6 pages)

8.1 Validation Against Literature

- Jacques et al. leaching stages: match/deviation

- Li et al. sulfate attack: ettringite → gypsum sequence
- Zuo et al. combined attack: LDH, Friedel, coupling effects

8.2 Pressure Effect Mechanism

- Interpretation of hydraulic pressure as renewal rate
- Physical analogy to electric migration (Li et al.)
- Quantitative difference (2-4× acceleration)

8.3 Fly Ash Influence

- Lower portlandite vs pure OPC
- Pozzolanic reaction buffering
- Aluminum availability for AFm/AFt
- LDH formation potential

8.4 Degradation Mechanisms

- Decalcification (Ca^{2+} leaching)
- Chloride binding (Friedel, C-S-H)
- Sulfate attack (expansion, gypsum)
- Coupled effects (neutralization, AFm competition)

9. Sensitivity and Limitations (2-3 pages)

- Parameter sensitivity (FA content, renewal rate, initial CH)
- Model assumptions:
    - Thermodynamic equilibrium (no kinetics)
    - Homogeneous paste (no spatial gradients)
    - Perfect mixing in pore solution
- Limitations:
    - No mechanical damage feedback
    - No transport coupling (1D diffusion)
    - Fly ash glass simplified as reactive oxide
- Recommendations for future work

10. Conclusions (1-2 pages)

- Summary of 6 scenarios
- Key findings (portlandite loss, Friedel, ettringite, pressure effect)
- Practical implications for durability
- Validation status vs experimental data

11. References (2-3 pages)

- Jacques et al. (2010), Li et al. (2020), Zuo et al. (2023)
- Lothenbach et al. (2019) CEMDATA18
- GEMS-PSI documentation
- Client's XRD/TGA data sources

12. Appendices (5-10 pages)

- Appendix A: Material input data (full tables)
- Appendix B: GEMS input files (JSON samples)
- Appendix C: Complete phase evolution tables
- Appendix D: Pore solution tables
- Appendix E: Python script examples

Supplementary Files

In `./deliverables/`:

1. `model_parameters.xlsx` - All input parameters in spreadsheet
2. `phase_data_all_scenarios.xlsx` - Complete numerical results
3. `calibration_details.pdf` - Extended validation documentation
4. `code_archive.zip` - All Python scripts and configuration files

Success Criteria

- [ ] Report is comprehensive (30-50 pages)
- [ ] All figures high-quality (300 dpi)
- [ ] Results reproducible from provided scripts
- [ ] Validation against client data documented
- [ ] Literature comparison thorough
- [ ] Assumptions and limitations clearly stated

Complete File Structure

```
CementFlyAsh_SaltAttack_Project/
│
├── gems_project/
│   ├── project_config.json
│   └── cemdata18.dch (GEMS database)
│
```

```
├── materials/
│   ├── cement_composition.json
│   ├── flyash_composition.json
│   └── material_validation.csv
│
├── recipes/
│   └── paste_recipe.json
│
├── solutions/
│   └── external_solutions.json
│
├── process_config/
│   └── process_parameters.json
│
├── scripts/
│   ├── hydration_28d.py
│   ├── run_PW_immersion.py
│   ├── run_PW_pressure.py
│   ├── run_NaCl_immersion.py
│   ├── run_NaCl_pressure.py
│   ├── run_mixed_immersion.py
│   ├── run_mixed_pressure.py
│   ├── plot_phase_evolution.py
│   ├── plot_pore_solution.py
│   ├── calculate_metrics.py
│   └── sensitivity_analysis.py
│
├── outputs/
│   ├── baseline_28d.json
│   ├── baseline_phases.csv
│   ├── baseline_poresolution.csv
│   │
│   ├── PW_Immersion/
│   │   ├── raw_results.json
│   │   ├── phase_evolution.csv
│   │   └── pore_solution_evolution.csv
│   │
│   ├── PW_Pressure/
│   │   └── (same structure)
│   │
│   ├── NaCl_Immersion/
│   │   └── (same structure)
│   │
│   ├── NaCl_Pressure/
│   │   └── (same structure)
│   │
│   ├── Mixed_Immersion/
│   │   └── (same structure)
│   │
│   ├── Mixed_Pressure/
│   │   └── (same structure)
│   │
│   ├── plots/
│   │   ├── portlandite_comparison.png
│   │   ├── sulfate_phases.png
│   │   ├── chloride_binding.png
│   │   ├── pH_evolution.png
│   │   ├── calcium_leaching.png
│   │   └── csh_decalcification.png
│   │
│   ├── summary_metrics.csv
│   └── stage_boundaries.json
│
├── validation/
│   ├── calibration_log.md
│   ├── sensitivity_results.csv
│   ├── sensitivity_CH_loss.png
│   └── validation_plots/ (XRD vs model)
│
├── deliverables/
│   ├── FINAL_REPORT.pdf
│   ├── model_parameters.xlsx
│   ├── phase_data_all_scenarios.xlsx
│   ├── calibration_details.pdf
│   └── code_archive.zip
│
├── docs/
│   ├── README.md (this file)
│   ├── QUICK_START.md
│   └── TROUBLESHOOTING.md
│
└── references/
├── Jacques_2010_Leaching.pdf
├── Li_2020_SulfateAttack.pdf
├── Zuo_2023_CombinedAttack.pdf
└── CEMDATA18_Documentation.pdf
```

Quick Start Guide

Day 1: Setup (2-3 hours)

1. Install GEMS/Cem-GEMS and Python environment
2. Verify database access (CEMDATA18)
3. Create project directories
4. Prepare material composition files

Day 2-3: Baseline (4-6 hours)

1. Run 28-day hydration (`hydration_28d.py`)
2. Compare with client XRD/TGA
3. Calibrate hydration degrees (iterate if needed)
4. Document baseline in `calibration_log.md`

Day 4-5: Process Implementation (8-10 hours)

1. Set up external solutions and process parameters
2. Implement and test one scenario (`run_PW_immersion.py`)
3. Replicate for remaining 5 scenarios
4. Run all simulations (5-30 min each)

Day 6: Validation (3-4 hours)

1. Check degradation trends vs literature
2. Run sensitivity analysis
3. Adjust parameters if outliers detected
4. Finalize calibration

Day 7-8: Analysis (6-8 hours)

1. Generate all comparison plots
2. Calculate derived metrics
3. Interpret mechanisms (stages, pressure effect)
4. Prepare figure captions

Day 9-12: Report Writing (16-24 hours)

1. Write methodology sections (thermodynamics, materials, process)
2. Present results with figures and tables
3. Discussion (validation, mechanisms, limitations)
4. Executive summary and conclusions
5. Compile appendices
6. Final review and formatting

Day 13: Delivery

1. Generate PDF report
2. Package supplementary files
3. Create code archive
4. Submit deliverables to client

Total effort: ~50-70 hours over 2 weeks

Troubleshooting

GEMS Convergence Issues

Problem: Equilibrium calculation fails to converge
Solutions:

- Reduce step size (smaller water/step)
- Increase solver iterations in GEMS settings
- Check for negative amounts in input (bug indicator)
- Simplify system (suppress minor phases temporarily)

Phase Not Forming

Problem: Expected phase (e.g., Friedel) doesn't precipitate
Solutions:

- Verify phase is in CEMDATA18 and not suppressed
- Check stoichiometry (Cl/AFm ratio sufficient?)
- Increase step resolution (more frequent equilibrations)
- Review pore solution supersaturation

Unrealistic Degradation

Problem: Portlandite disappears too fast or too slow
Solutions:

- Adjust renewal rate (water/step parameter)
- Re-calibrate baseline hydration degree
- Check external solution concentration
- Compare cumulative water with Jacques et al. benchmark

Python Script Errors

Problem: `xgems` import fails or JSON parsing error
Solutions:

- Reinstall xgems: `pip install --upgrade xgems`
- Validate JSON syntax with linter
- Check file paths (relative vs absolute)
- Use Python 3.8+ (earlier versions not supported)

Key Success Metrics

By project completion, you should have:

- [x] GEMS project with validated baseline (28d) matching XRD/TGA
- [x] 6 complete scenario simulations with converged results
- [x] Phase evolution plots showing clear degradation trends
- [x] Pore solution chemistry explaining mechanisms
- [x] Quantified pressure effect (2-4× acceleration factor)
- [x] Calibration documented with sensitivity analysis

- [x] 30-50 page report with methodology and discussion
- [x] All data files, scripts, and plots organized and archived
- [x] Results consistent with Jacques, Li, Zuo literature

Support and Resources

Documentation

- GEMS tutorial: https://cemgems.org/tutorial/getting-started/
- CEMDATA18 paper: Lothenbach et al. (2019)
- xGEMS Python docs: https://xgems.readthedocs.io

Key References

1. Jacques et al. (2010) *Cement Concrete Res.* - Leaching model methodology
2. Li et al. (2020) *Cement Concrete Comp.* - Sulfate attack mechanisms
3. Zuo et al. (2023) *J. Build. Mater.* - Combined NaCl + $MgSO_4$ thermodynamics

Contact

For project-specific questions, refer to client communication logs and experimental data provided in initial task documents.

Good luck with your modeling project!