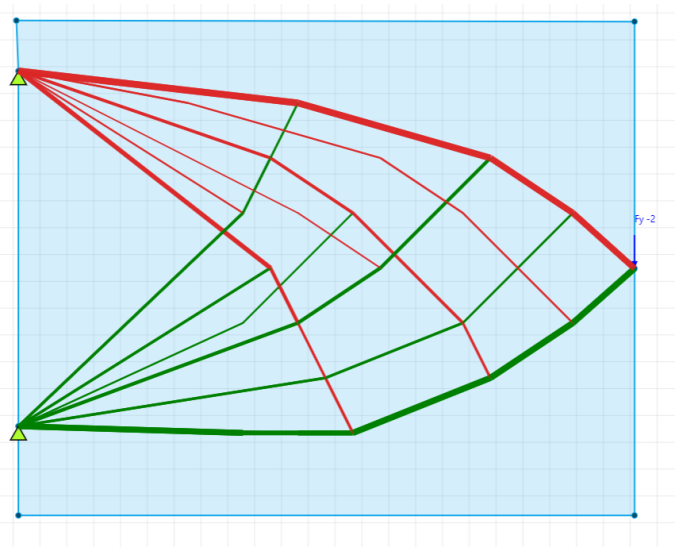
TopoKALC

Ground-Structure Topology Optimisation and Structural Load-Path Exploration



This document describes the engineering formulation implemented by TopoKALC, its role within the RKALC structural design workflow, and the interpretation of topology optimisation results for structural form exploration and strut-and-tie model development.

Document Revision

Revision Title	Rev #	Issue Date	By
First Issue	0	10th August 2026	RK

Table of Contents

1. Introduction

1.1. Overview

TopoKALC is a two-dimensional structural topology optimisation application developed within the RKALC engineering platform. Its purpose is to assist structural engineers in identifying efficient internal load-transfer mechanisms within an engineer-defined structural domain before the final structural idealisation has necessarily been selected.

Conventional structural analysis normally begins after the engineer has chosen the structural system. Beams, braces, walls, struts, ties and support conditions are defined first, and the subsequent analysis determines how that assumed structure responds. TopoKALC addresses an earlier design question: given a permitted region, applied actions and restraints, what load paths are mechanically efficient and capable of satisfying equilibrium?

This distinction is particularly important in discontinuity regions of reinforced concrete, transfer structures, deep beams, walls with openings, corbels, pile caps, irregular diaphragms, walking or discontinuous columns, and structural steel systems in which the preferred bracing form has not yet been established. In these cases, the force-transfer mechanism may be less obvious than the final strength calculations.

The current TopoKALC implementation uses a discrete ground-structure formulation. The user-defined domain is populated with potential structural nodes. Admissible axial members are generated between those nodes, an equilibrium optimisation problem is assembled, and a sparse linear-programming procedure determines a force-carrying subset. The result is presented as a force-weighted network of tension and compression paths.

1.2. Analysis Philosophy

TopoKALC is intended as an engineering reasoning tool rather than an automatic final-design generator. The topology is evidence about structural force flow. It is not, by itself, a completed reinforcement arrangement, a fully detailed steel framing system, or a code-verified strut-and-tie model.

The core engineering sequence is therefore:

Design domain + loads + restraints → admissible ground structure → equilibrium optimisation → dominant load paths → engineering idealisation

The final engineering interpretation remains with the structural engineer. This is consistent with published research on topology-assisted strut-and-tie modelling, where topology optimisation is used to generate or inform truss-like load paths while the detailed dimensioning of struts, ties and nodal zones remains a subsequent design task [1-4].

1.3. Scope of this Document

This document describes the methodology embodied in the current RKALC TopoKALC code. It distinguishes the implemented ground-structure method from continuum SIMP and ESO methods frequently discussed in topology-optimisation literature. It also explains how the current TopoKALC workflow complements the RKALC STM calculator by occupying the load-path discovery stage immediately before detailed strut-and-tie design.

Where a setting is transmitted by the user interface but is not currently active in the ground-structure optimisation equations, this document identifies that distinction explicitly. The objective is to describe the implemented engineering method rather than imply capabilities that are not presently used by the solver.

2. Engineering Basis of Topology Optimisation

2.1. Topology as a Structural Design Question

Topology optimisation seeks an efficient arrangement of structural material or load-carrying paths within an allowable region. Unlike conventional sizing optimisation, where the connectivity of the structure is fixed and only section sizes change, topology optimisation permits the structural connectivity itself to emerge from the optimisation problem.

Many continuum methods formulate the problem as minimum compliance or minimum strain energy subject to a material-volume constraint. SIMP methods represent each finite element using a continuous density variable, while evolutionary methods progressively remove inefficient material. These methods have produced truss-like structural forms and have been widely investigated for structural-concrete applications [1-4].

TopoKALC does not presently use a density-based continuum formulation. It instead adopts the ground-structure concept: a large set of possible truss members is generated first, and an optimisation problem determines which members are required to carry the loads efficiently. Ground-structure approaches have long been used for truss layout optimisation and have also been applied to automatic strut-and-tie model generation [1,2].

2.2. Relationship to Load Paths

A structural load path describes how applied actions are transferred through a structure to the restraints. In a truss idealisation, this path is represented explicitly by axial members and nodal equilibrium. The ground-structure method is therefore particularly suited to conceptual load-path exploration because the optimisation result is already expressed using structural objects familiar to engineers: nodes, members, force, tension and compression.

Published STM research similarly emphasises that stress trajectories or internal force flows may be condensed and idealised into straight compression and tension members. This is especially useful in D-regions, where conventional beam theory is not applicable and several statically admissible truss idealisations may exist [2,4].

2.3. Why a Ground-Structure Approach is Useful for RKALC

For an interactive web application, the ground-structure approach offers several practical advantages. The result is directly compatible with the language of strut-and-tie modelling; the equilibrium equations are sparse; arbitrary polygonal boundaries can be respected geometrically; and the engineer can control the discretisation and connectivity without requiring a continuum material interpolation model.

The method also provides a natural bridge between topology exploration and the RKALC STM calculator. TopoKALC can suggest dominant axial force trajectories; the engineer can then rationalise these trajectories into discrete struts, ties and nodes for detailed STM design.

3. Role of TopoKALC within RKALC

3.1. Upstream of the STM Calculator

Within the RKALC reinforced-concrete workflow, TopoKALC is positioned before detailed strut-and-tie analysis. Its principal purpose is to assist with one of the most judgement-dependent parts of STM design: establishing a rational internal load-transfer mechanism.

The integrated conceptual workflow is:

- Structural problem
- TopoKALC load-path discovery
- engineer rationalisation
- RKALC STM analysis and design

TopoKALC answers the question, "what efficient equilibrium paths are available within this region?" The RKALC STM calculator then answers the subsequent question, "can the engineer-defined struts, ties and nodal zones be dimensioned, reinforced and verified?"

3.2. Complementary Responsibilities

TopoKALC is concerned primarily with topology and equilibrium. The STM calculator is concerned with the detailed design model. The separation is deliberate. A numerical topology may contain several adjacent force paths that an engineer chooses to represent as one concrete compression field, or it may contain multiple tension trajectories that are rationalised into a practical reinforcement arrangement.

- TopoKALC: identifies candidate structural mechanisms and dominant tension/compression trajectories.
- Engineer: interprets, simplifies and rationalises the topology into a practical structural model.
- RKALC STM: performs detailed strut, tie and node analysis and design for the adopted idealisation.

4. Structural Problem Definition

4.1. Design Domain

The user defines a two-dimensional optimisation domain as a polygon. Polygon vertices may be created, edited and supplemented with additional vertices. The modelling interface supports snapping, polar drawing and direct manipulation so that the structural region can reflect the actual engineering geometry rather than a generic rectangular envelope.

Only the first optimisation domain is currently used by the server-side solver. The code architecture stores domains as a list, allowing future expansion, but the present solver passes the vertices of Domains(0) to mesh and member generation.

4.2. Supports

Supports are assigned at domain nodes. The front-end support object stores independent UX, UY and RZ restraint flags. In the current two-dimensional ground-structure optimisation, UX and UY restraints are assembled directly into the equilibrium problem. RZ is retained in the model object but axial ground-structure members have no rotational degree of freedom, so RZ does not currently contribute to the LP equilibrium equations.

The server protects support locations by explicitly inserting them as mesh nodes where they fall inside or on the optimisation boundary. This prevents the structural restraints from moving when the user changes the mesh resolution.

4.3. Point Loads

Point loads are applied to model nodes and contain horizontal and vertical components. The interface also records load-case indices. The current server-side topology solve assembles the point loads provided in the request into one global load vector; it does not presently solve the load cases independently. Load-case storage therefore supports the modelling interface and future extension, while the current optimisation should be interpreted according to the loads transmitted in the active request.

4.4. Input Validation

Before an optimisation job is created, the server requires a non-empty request, at least one domain, at least one support and at least one point load. These checks are engineering safeguards as well as software safeguards: a topology optimisation without a structural region, restraint or loading does not represent a meaningful load-transfer problem.

5. Domain Discretisation

5.1. Mesh Size as Design Resolution

The domain is discretised using a characteristic mesh size h. The mesh size controls the spacing between candidate structural nodes and therefore the spatial resolution at which a load path may change direction.

The current server constrains h to a range derived from the dimensions of the domain. The lower bound is related to the smaller domain dimension and protects the solver from excessively dense candidate structures. The upper bound prevents an excessively coarse mesh. The front-end further constrains the user slider using the short domain dimension so that the ground structure retains several useful rows across the narrow direction.

The user-facing engineering interpretation should therefore be "Design Resolution" rather than simply "Mesh Size". A finer resolution permits more detailed topology but increases the number of nodes and candidate members. A coarser resolution produces a more abstract mechanism and can be useful during early conceptual exploration.

5.2. Centred Staggered Grid

TopoKALC generates the internal mesh about the centre of the domain rather than exclusively from a lower-left origin. This reduces directional and positional bias when relatively coarse meshes are used.

Alternate rows are offset by half a mesh spacing. The staggered arrangement increases the variety of local member orientations compared with a purely orthogonal lattice and reduces some of the directional preference that would otherwise be imposed by the discretisation itself.

5.3. Protected and Boundary Nodes

The mesh generation procedure includes five categories of nodes: regular internal grid nodes, polygon vertices, boundary/grid intersection nodes, support nodes and point-load nodes. All are added uniquely within a numerical tolerance.

Boundary intersections are generated where polygon edges cross vertical or horizontal grid lines. This increases the fidelity with which the ground structure follows irregular boundaries. Supports and loads are inserted explicitly so that engineering boundary conditions remain exact geometric features rather than approximations to the nearest generic grid location.

6. Ground-Structure Generation

6.1. Candidate Members

After mesh generation, TopoKALC constructs a set of candidate axial members. Each pair of nodes is considered only when it lies within a prescribed connection range and the straight line connecting the nodes remains within the allowable polygon.

$$L_{ij} = \sqrt{[(x_j - x_i)^2 + (y_j - y_i)^2]}$$

The maximum permitted candidate length is

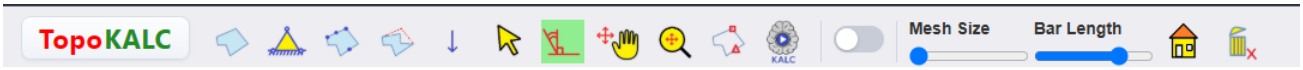
$$L_{max} = h\,\lambda$$

where h is the actual mesh spacing and λ is the user-controlled element-size or connectivity parameter. The implementation also imposes a minimum candidate length of approximately 0.55h to suppress undesirable very short members (see the mesh-size slider).

6.2. Load-Path Freedom

The parameter λ is better interpreted as "Load-Path Freedom" or "Bar length slider" than as a physical bar length. Increasing λ permits a node to connect to more distant nodes, allowing the optimisation to consider longer and more direct trajectories. Reducing λ forces load transfer to occur through more local connections.

The two main topology controls therefore have distinct engineering meanings: mesh size determines where the path may turn, while connection reach determines how far the path may travel between turns.



6.3. Spatial Binning

Candidate generation can become expensive if every node is tested against every other node. TopoKALC reduces this cost by placing nodes into spatial bins of approximately one mesh spacing. For each source node, only neighbouring bins within the connection search radius are examined. Distance checks and geometric admissibility tests are then applied to the reduced local set.

This implementation detail is important to interactive engineering use because a richer ground structure increases the number of possible structural mechanisms but also increases computational demand.

6.4. Candidate Limit

The current implementation rejects ground structures containing more than 25,000 candidate members. This is an operational safeguard. When the limit is exceeded, the engineer is instructed to increase the mesh size or reduce the connection reach. The message should be understood as a request to reduce the design-space resolution, not as a structural failure of the physical domain.

7. Geometric Admissibility of Candidate Load Paths

7.1. Why Endpoint Checks are Insufficient

For concave polygons, two candidate endpoints can both lie inside the structural domain while the straight segment between them passes outside the domain. A simple endpoint, midpoint or quarter-point check can therefore admit physically impossible members that cross openings, re-entrant corners or excluded regions.

7.2. Interval-Based Segment Test

TopoKALC uses a more robust segment-in-polygon procedure. The candidate segment is intersected with every polygon edge. The resulting intersection parameters divide the candidate member into intervals. A midpoint from every interval is then tested to confirm that it lies inside or on the polygon.

A candidate member is accepted only when both endpoints belong to the domain and every interval formed by boundary intersections remains admissible. Collinear intersections with polygon edges are also considered.

This procedure is particularly important for walls with openings, notches, irregular transfer regions and other non-convex geometries, because the optimiser must not be allowed to create a structurally impossible shortcut through excluded space.

8. Structural Stability and Connectivity

8.1. Ground-Structure Graph

Before the linear programme is solved, TopoKALC checks whether the generated candidate network provides a plausible structural route from the loaded nodes to the restraints. An adjacency graph is built from the candidate members, with one graph vertex per mesh node.

8.2. Reachability from Supports

Nodes associated with translational restraints are identified. A breadth-first traversal is then performed through the candidate-member graph, starting from all restrained nodes. Every non-zero loaded node must lie within this reachable structural network.

If a load node is isolated from the support network, the selected discretisation cannot provide an equilibrium mechanism. TopoKALC reports the ground structure as unstable and asks the engineer to reduce the mesh size.

8.3. Local Connectivity at Loaded Nodes

A further local check is performed at loaded nodes. A non-supported loaded node with fewer than two candidate connections is considered vulnerable to a coarse-mesh mechanism. This is a practical stability screen rather than a complete kinematic proof, but it catches common cases where an apparently valid domain has been discretised too coarsely to transmit the applied action.

9. Equilibrium and Optimisation Formulation

9.1. Member Direction

For a candidate member between nodes 1 and 2, define

$$\Delta x = x_2 - x_1, \quad \Delta y = y_2 - y_1, \quad L = \sqrt{(\Delta x^2 + \Delta y^2)}$$

$$c_x = \Delta x / L, \quad c_y = \Delta y / L$$

A unit axial force in the member contributes the following nodal equilibrium components:

$$[\, +c_x \;\; +c_y \;\; -c_x \;\; -c_y \,]^T$$

for the positive axial direction. The opposite direction is represented by the sign-reversed column.

9.2. Free Degrees of Freedom

The global model has two translational degrees of freedom per mesh node. UX and UY support restraints are mapped to the nearest protected mesh node and removed from the active equilibrium equations. The remaining free translational degrees of freedom form the rows of the LP equilibrium matrix.

9.3. Tension and Compression Variables

Each geometric candidate member is represented using two non-negative optimisation variables. One variable represents positive axial action and the second represents negative axial action.

$$q_i = q_i^{+} - q_i^{-}$$

$$q_i^+ \geq 0, \quad q_i^- \geq 0$$

This allows a standard non-negative LP variable representation while preserving the physical interpretation of tension and compression. After solution, the two contributions are combined to recover one signed force for each geometric member.

9.4. Objective Function

The current TopoKALC LP assigns each positive and negative member variable a cost equal to the geometric member length. Conceptually, the optimisation therefore solves

$$minimise \quad \Sigma_i L_i (q_i^+ + q_i^-)$$

subject to

$$A (q^+ - q^-) = -P$$

and non-negativity of q+ and q-.

The objective can be interpreted as a length-weighted axial-force measure. For equal force magnitude, a longer load path is more expensive than a shorter path. The optimiser therefore seeks a statically admissible network that balances force magnitude and path length.

The result should not be described as a continuum minimum-compliance or SIMP solution because the current code does not use element densities, elastic stiffness interpolation or a material-volume constraint in the active LP. It is a discrete ground-structure equilibrium optimisation.

9.5. Current Status of Other Settings

The request object currently contains VolumeFraction, Penalty, FilterRadius, Thickness and Material in addition to MeshSize and ElementSize. In the attached ground-structure solver, MeshSize and ElementSize actively control mesh and connectivity. The other settings are transmitted and retained for wider application architecture or future development but are not presently used in the LP objective or equilibrium constraints described above.

Accordingly, the present methodology should not claim that volume fraction, SIMP penalisation, filter radius, material modulus or section thickness influence the current ground-structure result unless and until those quantities are introduced into the active optimisation formulation.

10. Sparse Numerical Solution

10.1. Sparse Column Representation

Every axial member contributes to at most four translational equilibrium coefficients before support reduction. The equilibrium matrix is therefore sparse. TopoKALC stores each optimisation variable as a SparseLpColumn containing only the active row indices and non-zero coefficients associated with that candidate action.

This avoids constructing a dense matrix whose overwhelming majority of entries would be zero.

10.2. Scaling

Before the sparse LP is solved, equilibrium rows are scaled using the larger of the row coefficient magnitude and load-vector magnitude. Objective coefficients are similarly scaled by the maximum member cost. The purpose is numerical conditioning: the barrier solver performs more reliably when coefficient magnitudes are of comparable order.

10.3. Dual Barrier Procedure

The implemented solver uses a barrier formulation in the dual space. It maintains positive dual slacks, evaluates the barrier gradient, and applies Newton-like updates. The normal equations associated with each update are not formed as a dense matrix. Instead, matrix-vector products are evaluated directly from the sparse column data.

10.4. Preconditioned Conjugate Gradient

The normal-equation step is solved iteratively using a preconditioned conjugate-gradient procedure with a diagonal preconditioner assembled from the sparse columns. This reduces the computational burden compared with repeated dense factorisation for the problem sizes encountered in interactive ground-structure optimisation.

The engineering importance of the sparse implementation is that the optimiser can explore relatively rich structural networks while remaining suitable for a browser-based modelling workflow.

11. Recovery and Rationalisation of the Topology

11.1. Recovering Signed Member Forces

After the LP variables are solved, contributions from the positive and negative columns associated with each geometric member are summed to recover a signed axial force. Positive force is classified as tension and negative force as compression.

The result object also records member length and a length-weighted force quantity labelled Energy, calculated as |F|L. In the current code this value is a reporting quantity and is not used as a separate iterative optimisation criterion.

11.2. Initial Force Threshold

Very small solved forces are removed before the final topology is assembled. The current implementation retains members whose absolute force exceeds the larger of a small absolute tolerance and approximately 2% of the maximum solved member force.

This threshold is a topology-interpretation device. It prevents extremely small numerical force paths from dominating the graphical output while preserving members that materially participate in the equilibrium mechanism.

11.3. Removal of Stray Branches

A second cleanup stage removes weak dangling members. The algorithm first defines weak and strong force bands relative to the maximum retained force. Load and support nodes are protected. Weak members connected to an unprotected node of degree one are iteratively deleted, while strong members are retained.

The process repeats because removing one dangling member can expose another. This is intended to simplify the displayed structural mechanism without severing the essential load-to-support network.

11.4. Visual Force Hierarchy

The front-end result renderer scales line thickness using a non-linear function of the member force ratio. Dominant load paths therefore appear visibly stronger than secondary paths. Compression and tension are rendered differently so that the engineer can distinguish the two behaviours immediately.

The graphical result is therefore not only a connectivity diagram. It is a qualitative force-flow map showing the hierarchy of the optimised axial mechanism.

12. Engineering Interpretation of the Controls

12.1. Design Resolution

The mesh-size control should be interpreted as Design Resolution. Decreasing the mesh spacing creates more potential turning points and permits a finer representation of load flow. Increasing it produces a simpler, more abstract network and generally reduces computation time.

A coarse solution is not necessarily less useful at concept stage. It may reveal the dominant structural mechanism more clearly. A refined solution is valuable when the engineer wants to test whether that mechanism remains stable under increased geometric freedom.

12.2. Load-Path Freedom

The element-size control should be interpreted as Load-Path Freedom or Connection Reach. It is a multiplier applied to the actual mesh spacing, not a direct physical member length. Increasing it allows longer candidate connections and can reveal more direct diagonal or spanning trajectories. Reducing it encourages the solution to pass through local intermediate nodes.

12.3. Recommended Workflow with the Controls

1. Start with a moderate or coarse Design Resolution to reveal the principal mechanism quickly.
2. Use moderate Load-Path Freedom so the ground structure contains diagonal alternatives without becoming unnecessarily dense.
3. Inspect whether the dominant load paths are mechanically credible and correspond to expected reactions.
4. Refine Design Resolution and rerun the problem.
5. Confirm whether the same primary structural mechanism remains visible.
6. Adjust Load-Path Freedom where long direct paths appear artificially restricted or where the candidate structure becomes unnecessarily dense.
7. Use the stable recurring topology as the basis for engineering rationalisation.

13. Application to Structural Concrete and RKALC STM

13.1. D-Regions and the STM Problem

Strut-and-tie modelling is especially important in D-regions, where discontinuities in geometry or loading invalidate the simple assumption of a linear strain distribution. Published literature identifies the selection of an appropriate ST model as one of the most difficult parts of the design process, particularly for complex geometry and loading [1-4].

Topology optimisation is relevant because the internal load-transfer mechanism can often be represented by a truss-like network. Research has therefore investigated both ground-structure linear-programming approaches and continuum evolutionary methods for automatic or assisted STM generation [1-4].

13.2. TopoKALC as the Load-Path Discovery Stage

Within RKALC, TopoKALC is not intended to replace the STM calculator. It fills the stage immediately before detailed STM design. The engineer defines the D-region geometry, loading and restraints, obtains a tension/compression topology, and then rationalises the dominant trajectories into discrete struts, ties and nodal zones.

D-region definition → TopoKALC → topology interpretation → STM idealisation → RKALC STM design

13.3. Compression Paths

Dominant negative axial-force trajectories can be interpreted as candidate compression paths. In reinforced concrete these may suggest the centreline of concrete struts or the principal direction of a broader compression field. Several adjacent compression paths should not automatically be converted into several physical struts; engineering rationalisation is required.

13.4. Tension Paths

Dominant positive axial-force trajectories can indicate regions in which tensile force must be carried. For reinforced concrete, these paths can inform the location and direction of reinforcement ties. Practical detailing, anchorage, curtailment and distributed reinforcement remain subsequent design considerations.

13.5. Nodal Zones

Convergence and redirection of dominant force paths can indicate locations at which an engineer may establish STM nodes. The numerical mesh node itself is not automatically a physical nodal zone. The engineer must define the nodal geometry according to the adopted struts, ties, bearing regions and reinforcement anchorage.

13.6. Independent Review of an Existing STM

TopoKALC can also be used as an independent review tool. An engineer may reproduce the domain, actions and restraints corresponding to an existing STM and compare its principal struts and ties with the dominant optimised paths. Agreement provides useful supporting evidence. Disagreement does not by itself prove that the existing STM is invalid, but it identifies a mechanism that deserves further engineering investigation.

14. Application to Structural Form Exploration

14.1. Beyond Reinforced Concrete

The ground-structure formulation is not inherently limited to concrete. Because the current optimisation is based on nodal equilibrium and axial load paths rather than a concrete constitutive model, the same mechanism can be used conceptually to investigate steel bracing, truss forms, transfer systems and other structural layouts.

The resulting lines should be interpreted as candidate force trajectories rather than automatically sized physical members. Buckling, connection design, stiffness compatibility, robustness, constructability and architectural constraints must be considered separately.

14.2. What-if Engineering

The interactive nature of TopoKALC makes it useful for sensitivity studies. An engineer can move a support, change the direction of a load, modify an opening or reshape the domain and observe whether the dominant force-transfer mechanism changes. This supports conceptual design decisions that are often difficult to obtain from one fixed finite element model.

15. Engineering Workflow

15.1. Model Creation

1. Draw the allowable structural domain.
2. Assign supports to the required geometric nodes and define UX/UY restraints.
3. Apply horizontal and vertical point loads.
4. Select an initial Design Resolution and Load-Path Freedom.

5. Run the topology optimisation.

15.2. Solver Workflow

1. Validate domain, supports and loading.
2. Generate the staggered internal mesh and protected nodes.
3. Generate admissible candidate members within the selected reach.
4. Confirm connectivity between loaded nodes and restrained nodes.
5. Assemble sparse tension and compression equilibrium columns.
6. Scale and solve the equality-constrained LP.
7. Recover signed axial forces.
8. Remove insignificant and weak dangling paths while protecting loads and supports.
9. Return nodes, members and force state for graphical interpretation.

15.3. Engineering Review

1. Identify the dominant compression and tension paths.
2. Check whether the paths provide a credible route from applied loads to reactions.
3. Rerun at a different resolution to test topology stability.
4. Rationalise multiple numerical paths into practical structural members or stress fields.
5. For reinforced concrete, transfer the rationalised model into RKALC STM for detailed design.
6. For steel or other structural forms, develop the selected concept into a conventional analysis/design model.

16. Pseudocode of the TopoKALC Mechanism

16.1. Overall Procedure

```
PROCEDURE RunTopoKALC(problem)
    VALIDATE problem has domain, supports and loads

    mesh <- GenerateTopoMesh(problem)
    members <- GenerateCandidateMembers(mesh, domain, settings)

    IF members.count exceeds operational limit THEN
        REPORT that design space is too dense
    END IF

    ValidateGroundStructureStability(mesh, members, supports, loads)

    result <- SolveGroundStructureLP(mesh, members, supports, loads)

    RETURN result
END PROCEDURE
```

16.2. Mesh Generation

```
PROCEDURE GenerateTopoMesh(problem)
    READ polygon vertices
    DETERMINE domain bounds, width, height
    SELECT mesh spacing h within allowed range
    SELECT connection parameter lambda

    COMPUTE domain centre cx, cy

    FOR each staggered mesh row about domain centre
        IF row number is alternate THEN
            offset row by h/2
        END IF

        FOR each grid position on row
            IF point is inside or on polygon THEN
                ADD unique mesh node
            END IF
```

```
        END FOR
    END FOR

    ADD all polygon vertices as protected nodes
    ADD polygon/grid boundary intersection nodes
    ADD support locations as protected nodes
    ADD load locations as protected nodes

    RETURN mesh
END PROCEDURE
```

16.3. Candidate Ground Structure

```
PROCEDURE GenerateCandidateMembers(mesh, polygon, settings)
    maxLength <- meshSpacing * connectionReach
    minLength <- 0.55 * meshSpacing

    PLACE mesh nodes into spatial bins

    FOR each node n1
        SEARCH neighbouring bins within reach

        FOR each candidate node n2
            SKIP duplicate node pair
            COMPUTE squared distance
            REJECT if shorter than minLength
            REJECT if longer than maxLength
            REJECT if segment leaves polygon

            ADD candidate axial member n1-n2
        END FOR
    END FOR

    RETURN candidate members
END PROCEDURE
```

16.4. Segment Admissibility

```
FUNCTION SegmentInsidePolygon(p1, p2, polygon)
    REJECT unless both endpoints belong to domain

    cuts <- {0, 1}

    FOR each polygon edge
        FIND intersection parameter t with candidate segment
        ADD valid t to cuts
        IF candidate and edge are collinear THEN
            ADD projected edge-end parameters
        END IF
    END FOR

    SORT and remove duplicate cuts

    FOR each interval between consecutive cuts
        midpoint <- point at mean interval parameter
        IF midpoint is outside polygon THEN
            RETURN FALSE
        END IF
    END FOR

    RETURN TRUE
END FUNCTION
```

16.5. Connectivity Check

```
PROCEDURE ValidateGroundStructureStability(mesh, members, supports, loads)
    BUILD adjacency graph from candidate members
    IDENTIFY nodes with translational restraint

    IF no restraint nodes exist THEN
        REPORT no effective restraint
    END IF

    reachable <- breadth-first traversal from all restraint nodes
```

```
    FOR each non-zero loaded node
        IF loaded node is not reachable THEN
            REPORT unstable/coarse ground structure
        END IF

        IF loaded node is not a support AND local degree < 2 THEN
            REPORT unstable/coarse ground structure
        END IF
    END FOR
END PROCEDURE
```

16.6. LP Assembly and Solution

```
PROCEDURE SolveGroundStructureLP(mesh, members, supports, loads)
    BUILD set of restrained translational DOFs
    ASSEMBLE global nodal load vector P
    BUILD list of free DOFs
    b <- -P at free DOFs

    FOR each candidate member i
        COMPUTE length Li and direction cosines cx, cy

        CREATE positive axial column
            equilibrium coefficients = [+cx,+cy,-cx,-cy]
            objective cost = Li

        CREATE negative axial column
            equilibrium coefficients = [-cx,-cy,+cx,+cy]
            objective cost = Li
    END FOR

    SCALE equilibrium rows and objective costs
    x <- SolveSparseEqualityLP(columns, b)

    FOR each geometric member i
        Fi <- sum(sign * solved variable for member i)
    END FOR

    REMOVE forces below significance tolerance
    CLASSIFY Fi >= 0 as tension; Fi < 0 as compression
    REMOVE weak dangling paths while protecting load/support nodes

    RETURN topology result
END PROCEDURE
```

16.7. Sparse Barrier Solver

```
PROCEDURE SolveSparseEqualityLP(columns, b)
    INITIALISE dual vector y = 0
    INITIALISE barrier parameter mu

    REPEAT for decreasing values of mu
        REPEAT Newton iterations
            slack_j <- cost_j - column_j^T y
            REQUIRE all slacks > 0

            COMPUTE barrier gradient g
            COMPUTE diagonal weights d_j = mu / slack_j^2

            SOLVE sparse normal equation for dy
                using preconditioned conjugate gradient

            LINE SEARCH for a step preserving positive slacks
            UPDATE y
        UNTIL gradient sufficiently small

        REDUCE mu
    UNTIL barrier parameter sufficiently small

    RECOVER primal variables x_j = mu / slack_j
    RETURN x
END PROCEDURE
```

17. Numerical Behaviour, Mesh Sensitivity and Limitations

17.1. Mesh Dependence

Topology optimisation is inherently sensitive to the available design space. Ground-structure methods are particularly affected by the node layout and candidate connectivity because the optimiser can only select paths that have been generated. Published STM topology literature explicitly notes that the ground-structure grid can significantly influence the optimal topology, and that overly sparse ground structures may fail to represent the underlying continuum behaviour [1,2].

TopoKALC addresses this practically through staggered mesh generation, adjustable design resolution and adjustable connection reach. These measures reduce but do not eliminate discretisation dependence. Engineering users should therefore confirm important load paths by rerunning the model at different reasonable resolutions.

17.2. No Continuum Stress Field

The current TopoKALC method does not calculate a plane-stress continuum field and then trace principal stresses. It also does not use SIMP densities or ESO material removal. The topology is obtained directly from an axial ground structure satisfying nodal equilibrium. This makes the result well suited to truss-like interpretation, but the user should not describe it as the unique continuum stress trajectory of the original material body.

17.3. No Automatic Member Sizing

The current optimisation cost uses member length and axial-force variables; it does not simultaneously design real cross-sectional areas, concrete strut widths, steel reinforcement areas or buckling capacities. Result members therefore indicate load paths and relative force importance rather than final physical sizes.

17.4. Post-Processing is Interpretive

Force thresholds and stray-member cleanup improve readability, but they are post-processing operations. The displayed topology is therefore a rationalised representation of the numerical solution. For critical work, the engineer should retain awareness that small secondary paths may have been suppressed from the visual model.

17.5. Load Cases

The current front end stores load-case indices. The attached server-side ground-structure solve assembles the supplied point loads into one load vector rather than performing a separate multi-load-case optimisation. A future multi-load-case formulation could require candidate members to satisfy several equilibrium systems simultaneously or combine objective contributions across cases.

17.6. Two-Dimensional Idealisation

The current TopoKALC implementation is two-dimensional. Out-of-plane force transfer, torsion, three-dimensional node behaviour and spatial bracing are outside the present solver formulation.

18. Current Implementation Scope

18.1. Implemented

- Arbitrary 2D polygonal optimisation domain.
- Interactive domain editing, snapping, supports and point loads.
- Centred staggered mesh generation.
- Boundary, load and support node protection.

- Ground-structure generation with spatial binning.
- Full segment-inside-polygon admissibility test.
- Graph-based connectivity and loaded-node stability checks.
- Sparse equilibrium LP with separate tension and compression variables.
- Length-weighted objective function.
- Scaling, dual barrier iterations and PCG normal-equation solution.
- Relative force filtering and dangling-branch cleanup.
- Tension/compression graphical display and force-weighted line width.
- Local browser persistence of domain, support and load data.

18.2. Presently Reserved or Inactive in the Ground-Structure Solve

- VolumeFraction: present in TopoSettings but not currently used as a volume constraint.
- Penalty: present in TopoSettings but not currently used as a SIMP penalisation exponent.
- FilterRadius: present in TopoSettings but not currently used as a continuum sensitivity/density filter.
- Thickness: stored with the domain/settings but not currently used in the LP objective or equilibrium.
- Material: stored with the domain/settings but no elastic modulus or material strength currently enters the LP.
- RZ support restraint: stored in the model but not active in the translational axial ground-structure equations.
- Separate multi-load-case topology optimisation: load-case data exists in the interface, but the attached solve assembles the supplied loads into one equilibrium vector.

19. Recommended Engineering Use

19.1. Conceptual Design

Use TopoKALC to reveal structural mechanisms before committing to a detailed member arrangement. Begin with a coarse or moderate resolution and focus on the primary route between load introduction and support reactions.

19.2. STM Development

For reinforced-concrete D-regions, use TopoKALC to identify candidate compression and tension trajectories. Rationalise these into a simple statically admissible model, then complete the design in RKALC STM. Do not transfer every numerical member directly into the final STM.

19.3. Sensitivity and Peer Review

Use repeated analyses to test how sensitive a proposed mechanism is to mesh resolution, support position, load direction or geometric openings. Use the result as an independent check on an engineer-generated load path, not as an automatic declaration of correctness.

19.4. Engineering Judgement

A mechanically efficient topology may still be impractical. Detailing, anchorage, minimum reinforcement, concrete compression capacity, buckling, connection geometry, robustness, serviceability, fire, durability and construction sequence can all govern the final structure. TopoKALC informs these decisions; it does not replace them.

20. Conclusions

TopoKALC provides RKALC with an interactive ground-structure topology optimisation capability focused on structural load-path discovery. The method begins from an engineer-defined polygonal domain, loading and

restraints, creates a discretised set of admissible axial paths, enforces nodal equilibrium, and solves a sparse length-weighted optimisation problem to identify a dominant tension/compression mechanism.

The implementation places particular emphasis on engineering usability: loads and supports are protected during meshing, candidate paths are prevented from leaving irregular domains, unstable coarse ground structures are detected before optimisation, and the final network is simplified into an interpretable force hierarchy.

Within the RKALC ecosystem, the most important application is its relationship with the STM calculator. TopoKALC addresses the question of structural idealisation - where the important compression and tension paths are likely to occur - while RKALC STM addresses detailed design of the rationalised struts, ties and nodal zones.

The resulting workflow allows engineers to move from geometry and loading, through computational load-path exploration, to an engineer-defined and verifiable structural model. This makes topology optimisation a practical part of conceptual structural engineering rather than an isolated optimisation exercise.

21. References

[1] Liang, Q.Q., Xie, Y.M. and Steven, G.P. (2000). Topology Optimization of Strut-and-Tie Models in Reinforced Concrete Structures Using an Evolutionary Procedure. ACI Structural Journal, 97(2), 322-330.

[2] Kwak, H.-G. and Noh, S.-H. (2006). Determination of strut-and-tie models using evolutionary structural optimization. Engineering Structures, 28, 1440-1449.

[3] Kim, H. and Baker, G. (2002). Topology Optimization for Reinforced Concrete Design. Fifth World Congress on Computational Mechanics, Vienna.

[4] Bruggi, M. (2009). Generating strut-and-tie patterns for reinforced concrete structures using topology optimization. Computers & Structures, 87, 1483-1495.

[5] Sugianto, A. (2019). Evaluating strut-and-tie models for concrete elements by nonlinear finite element analysis. MSc Thesis, Delft University of Technology.

22. Appendix A - Detailed Pseudocode

22.1. A.1 Complete TopoKALC Flow

```
INPUT:
    polygon domain
    support locations and translational restraints
    point loads
    design resolution h
    connection reach lambda

VALIDATE INPUT
    if domain missing -> stop
    if supports missing -> stop
    if loads missing -> stop

GENERATE MESH
    calculate domain bounds
    clamp h to permitted range
    generate centre-origin staggered rows
    retain only points inside/on domain
    add polygon vertices
    add boundary/grid intersections
    add support nodes
    add load nodes

GENERATE GROUND STRUCTURE
    maxLen = h * lambda
    minLen = 0.55 * h
    bin nodes spatially
    for each node pair within search neighbourhood:
        reject duplicate pair
        reject if outside length range
        reject if full segment does not remain in polygon
        otherwise add candidate member

CHECK STRUCTURAL CONNECTIVITY
    build adjacency graph
    start traversal at restrained nodes
    require every loaded node to be reachable
    require reasonable local connectivity at loaded nodes

ASSEMBLE LOAD VECTOR
    two DOFs per mesh node: UX, UY
    map supports to restrained DOFs
    map point loads to nodal forces
    retain equilibrium rows only for free DOFs

ASSEMBLE LP COLUMNS
    for each candidate member:
        determine direction cosines cx, cy
        add tension variable with cost = member length
        add compression variable with cost = member length
        each variable contributes only to endpoint equilibrium rows

SCALE LP
    remove rows having no coefficients unless loaded
    scale rows by coefficient/load magnitude
    scale objective by maximum cost

SOLVE SPARSE EQUALITY LP
    initialise dual variables
    repeat barrier stages:
        compute positive slacks
        compute barrier gradient
        form normal-equation operator implicitly
        solve for search direction by PCG
        line-search while preserving positive slack
        reduce barrier parameter
    recover non-negative primal variables

RECOVER TOPOLOGY
    combine tension/compression variables into signed member force
    calculate maximum force
    discard very small force members
    label remaining members as tension/compression

CLEAN RESULT
```

```
    identify load/support nodes as protected
    retain strong paths
    iteratively remove weak dangling branches from unprotected nodes

OUTPUT
    mesh nodes
    retained members
    signed axial forces
    tension/compression state
    member lengths
    topology summary

DISPLAY
    scale line width by relative force magnitude
    colour tension and compression separately
    allow engineer to interpret/rationalise topology
```

22.2. A.2 Engineering Interpretation

```
ENGINEERING USE OF RESULT:

    IF purpose = reinforced concrete STM THEN
        identify dominant compression trajectories
        identify dominant tension trajectories
        rationalise groups into practical struts and ties
        establish nodal zones
        transfer rationalised model to RKALC STM
        complete strut, tie, node, anchorage and reinforcement checks

    ELSE IF purpose = structural form exploration THEN
        identify dominant axial paths
        propose bracing/truss/member arrangement
        build conventional structural analysis model
        check strength, buckling, serviceability and connections
    END IF

    repeat TopoKALC at modified resolution/boundary conditions
    confirm that the adopted engineering mechanism is robust
```