# Addressing Identified Issues in the SCP Workflow (Problems B, C, D)

## Mitigating Trust-Region Oscillations and Load Coupling

One key issue is the **oscillation of the objective (compliance)** during sequential convex programming iterations, especially when **design-dependent loads** (shell reactions) are updated each step. This can lead to negative trust-region ratios (ρ < 0) and rejected steps, hurting convergence. Recent research in sequential convex programming (SCP) has shown that such oscillations are an inherent phenomenon of trust-region methods: if the trust region is too rigid, the solution may bounce around the optimum (often called “solution chattering” or the **oscillation phenomenon**[[1]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=the%20trust%20region%20radius%20is,a%20soft%20constraint%20when%20oscillation)[[2]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=,optimality%20of%20the%20SCP%20method)). In fact, Xie *et al.* (2022) proved that oscillation is inherent to trust-region SCP for nonlinear problems and derived conditions when it occurs[[3]](https://www.researchgate.net/publication/358888469_Oscillation_Phenomenon_in_Trust-Region-based_Sequential_Convex_Programming_for_the_Nonlinear_Trajectory_Planning_Problem#:~:text=suitable%20for%20onboard%20aerospace%20applications,via%20a%20complex%20reentry%20trajectory)[[4]](https://www.researchgate.net/publication/358888469_Oscillation_Phenomenon_in_Trust-Region-based_Sequential_Convex_Programming_for_the_Nonlinear_Trajectory_Planning_Problem#:~:text=affecting%20the%20convergence,contribution%20of%20this%20paper). To **remedy** this, several techniques have been proposed:

* **Soft or Adaptive Trust Regions:** Instead of a hard trust-region constraint, treat it as a *soft* constraint or add it as a penalty term in the objective. This way, when oscillation is detected, the algorithm effectively relaxes the trust-region restriction[[5]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=oscillation%20phenomenon%20or%20solution%20chattering,trust%20region%20radius%20terms%20with)[[2]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=,optimality%20of%20the%20SCP%20method). For example, Xie *et al.* (2023) introduced a *higher-order soft trust-region* method that adds trust-region terms (of various orders) directly into the objective function[[2]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=,optimality%20of%20the%20SCP%20method). This approach improved convergence by allowing controlled constraint violations and guiding iterates more smoothly toward the optimum[[2]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=,optimality%20of%20the%20SCP%20method). A follow-up by Xie *et al.* (2024) combined multiple orders (“hybrid-order” trust regions) to further balance convergence and optimality[[2]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=,optimality%20of%20the%20SCP%20method).
* **Dynamic Penalty on Oscillation:** Another effective strategy is to augment the objective with an adaptive penalty term whenever oscillation is observed. Liu *et al.* (2024) proposed introducing a **quadratic penalty** to the objective if the objective value starts oscillating between iterations[[6]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=oscillation%20phenomenon%20or%20solution%20chattering,occurrence%20condition%20judgment%20is%20proposed)[[7]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=). Their method monitors the history of iteration improvements; once oscillation criteria are met, a penalty term is added (with a weight that updates based on iteration history) to damp the oscillation[[6]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=oscillation%20phenomenon%20or%20solution%20chattering,occurrence%20condition%20judgment%20is%20proposed)[[7]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=). This *variable quadratic penalty* essentially biases the next step to prevent back-and-forth toggling, and their results showed it **eliminated oscillations** and significantly improved SCP convergence[[8]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=function%20oscillation%20phenomenon%20occurs%2C%20the,the%20oscillation%20and%20improve%20convergence)[[9]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=To%20overcome%20the%20oscillation%2C%20the,the%20direction%20with%20the%20minimum).
* **Trust-Region Adjustment Policies:** Fine-tuning the trust-region update logic can also help. The current policy (shrink if ρ is low/negative, expand if ρ is high) can be augmented with oscillation detection. For instance, Xie *et al.* (2022) suggest detecting oscillation by looking at successive iteration changes; if the algorithm keeps shrinking the radius repeatedly or objective change signs, an oscillation is occurring[[10]](https://depts.washington.edu/uwrainlab/wordpress/wp-content/uploads/2020/01/ACC_2020.pdf#:~:text=it%2C%20followed%20by%20a%20simple,SEQUENTIAL%20CONVEX%20PROGRAMMING). In such cases, one might **halt further shrinkage** and instead switch strategy (e.g. apply a soft trust region or penalty as above). By processing the trust region in a softer manner when oscillation occurs, their approach avoided the stagnation or wild jumps that a strict trust-region could cause[[5]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=oscillation%20phenomenon%20or%20solution%20chattering,trust%20region%20radius%20terms%20with)[[2]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=,optimality%20of%20the%20SCP%20method).
* **Under-Relaxation of Load Updates:** In the specific context of the shell-truss coupling, one planned improvement is to **freeze or relax the loads** during the line search or trust-region step. This is analogous to techniques in partitioned multi-physics simulations where full updates can cause instability. Instead of immediately using the new shell reaction forces after every tiny design change, the algorithm can hold the load fixed for a few inner iterations or interpolate between the old and new loads (under-relaxation). While we did not find a specific paper on “freezing loads” in SCP, this idea is consistent with the broader practice of *simultaneous analysis and design (SAND)* in structural optimization. In a SAND approach, the design and state (loads, displacements) are solved in one unified formulation, which ensures consistency and can prevent divergent oscillations[[11]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=Solution%20procedures%20in%20structural%20optimization,In%20this%20paper%2C%20we%20make)[[12]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=alternately%20in%20an%20iterative%20scheme,These%20schemes%20give%20rise%20to). Larsson and Rönnqvist (1995), for example, showed that solving the analysis equations together with the design (via augmented Lagrangian methods) makes the dependency explicit and avoids the guess-update oscillation present in a nested update loop[[11]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=Solution%20procedures%20in%20structural%20optimization,In%20this%20paper%2C%20we%20make)[[13]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=conditioning%20inherent%20in%20penalty%20reformulations,are%20promising%20for%20future%20developments). Applying a similar idea here – effectively decoupling the load update frequency from every design step – could stabilize the convergence. In summary, **softening the trust-region constraints** and **smoothing the load updates** (either through penalty methods or partial updates) are techniques supported by recent research to resolve oscillation and improve convergence in sequential convex optimization.

## Handling Design-Dependent Loads in Structural Optimization

The interaction between the shell FE analysis and the truss optimization means the **loading on the truss is design-dependent** – as node angles $\theta$ change, the support locations and reaction forces from the shell change. Design-dependent loads are notoriously challenging in structural optimization because the load **magnitude, direction, or location changes as the design changes**[[14]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=,Those). This can lead to non-smooth objectives and difficulties in computing correct sensitivities, which in turn causes the oscillations discussed above. The literature on topology and shape optimization has several approaches to handle design-dependent loads, which could be informative here:

* **Continuous Load Interpolation:** One approach is to define the load or pressure field as a continuous function of the design variables, to avoid sudden changes. For example, Kumar and Langelaar (2019) treat pressure loads in topology optimization by introducing a fictitious intermediary (Darcy’s law fluid) so that pressure on a disappearing surface gradually tapers off instead of vanishing abruptly[[15]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=3D%20compliant%20mechanisms%20and%20loadbearing,a%20function%20of%20the%20design)[[14]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=,Those). They assign a continuous field that transitions from solid to fluid, ensuring the pressure load *evolves smoothly* with the topology. The pressure is converted to nodal forces via a consistent formulation, and importantly, **analytical sensitivities** of the load can be derived[[15]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=3D%20compliant%20mechanisms%20and%20loadbearing,a%20function%20of%20the%20design). This ensures the optimizer “knows” how loads move/change with design, rather than being surprised by abrupt load re-distributions.
* **Load Sensitivity and Co-Design:** In our problem, a simpler analog would be to incorporate the *sensitivity of the support reactions with respect to the node positions* into the optimization model. If the derivative $\partial \mathbf{F}\_{\text{truss}}/\partial \theta$ is available (perhaps via adjoint analysis of the shell FEA), one could include that in the gradient of compliance. This would make the predicted compliance change more accurate when $\theta$ moves, reducing the discrepancy between predicted and actual $\mathcal{C}$. Some advanced methods treat the reaction forces or other state variables as additional design variables with constraints enforcing equilibrium – essentially a **co-design optimization**. As noted, simultaneous analysis-design formulations impose the equilibrium constraints alongside design constraints, avoiding the lag between design update and load re-computation[[11]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=Solution%20procedures%20in%20structural%20optimization,In%20this%20paper%2C%20we%20make)[[16]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=An%20advantage%20of%20the%20simultaneous,first%20is%20a%20projected%20Newton). While more computationally intensive, such formulations guarantee that any intermediate design remains in equilibrium with some consistent load, thus preventing unphysical load spikes.
* **Partitioned Iteration with Relaxation:** If fully coupling the analysis is too complex, an alternative is the *partitioned iterative approach* but with relaxation. In fluid-structure optimization, for instance, one often updates the fluid and structure sequentially but uses an **under-relaxation factor** to update the loads (or displacements) gradually for stability[[17]](https://onlinelibrary.wiley.com/doi/full/10.1002/eng2.12514#:~:text=On%20stability%20and%20relaxation%20techniques,with%20the%20hydrodynamic%20condition). By analogy, after computing new shell reactions, one could update the truss load vector as $\mathbf{F}*{\text{truss}}^{new} = \alpha \mathbf{F}*$ with some $0<\alpha<1$. This smooths out large jumps in loading. Although we haven’t found a specific paper on this technique in truss-shell optimization, the approach is well-understood in multiphysics coupling[[17]](https://onlinelibrary.wiley.com/doi/full/10.1002/eng2.12514#:~:text=On%20stability%20and%20relaxation%20techniques,with%20the%20hydrodynamic%20condition)[[18]](https://www.researchgate.net/publication/327139340_Iterative_coupling_algorithms_for_large_multidomain_problems_with_the_boundary_element_method#:~:text=,the%20boundary%20element%20method) and would likely mitigate the sudden compliance “jumps” that violate trust-region predictions.}}^{\text{shell}} + (1-\alpha)\mathbf{F}\_{\text{truss}}^{old
* **Design-Dependent Load in Topology Optimization:** It’s worth noting that design-dependent load problems have been studied in topology optimization literature, as these share the issue of changing load application area. Researchers have developed specialized methods (level-set methods, material penalization approaches) to handle loads that move with the boundary[[19]](https://ui.adsabs.harvard.edu/abs/2018IJNME.115..825E#:~:text=Level%20set%20topology%20optimization%20for,to%20solve%20the%20compliance)[[14]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=,Those). For example, **Allaire et al. (2004)** and others treated pressure loads by updating the load as the boundary evolves, and **Sigmund & Peterson (1998)** introduced methods to handle follower forces. The general consensus is that naive approaches (treating load as fixed or instantly moving) fail, and one must either *introduce a continuous model for the load* or *optimize the load placement concurrently*. In summary, applying these insights: we should consider **making the load application a smooth function of the design** (or at least updating it in a controlled manner). This will ensure the SCP subproblem remains a valid approximation, thereby addressing problem “B” at its root cause.

## Incorporating Buckling and Stability Constraints (Problem C)

Another challenge (problem C) is how to account for **buckling constraints and structural stability** in the optimization. The current approach uses an **AASI buckling threshold** heuristic – essentially imposing a lower bound on cross-sectional area $A\_e$ if a member’s compressive force exceeds a critical value. While this is a practical engineering fix, it is not a rigorous or efficient way to enforce stability constraints. In literature, **truss optimization with buckling constraints** is known to be difficult because the critical buckling load is a non-convex, nonlinear function of the design (it depends on $A$ and the member length in a hyperbolic way)[[20]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=of%20bars%20and%20residual%20stresses,the%20critical%20buckling%20strength%20is)[[21]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=linearized%20iterative%20allowable%20stress%20method,the%20remarkable%20effect%20of%20including). Nevertheless, several techniques have been proposed to handle buckling in a more principled manner:

* **Local Euler Buckling Constraints:** Many works include **Euler buckling formulas** as constraints on each bar. For example, one can require $\frac{P\_e}{A\_e} \le \sigma\_{\text{allow}}$ where $P\_e = \pi^2 E I / L^2$ is the Euler buckling load (with $I$ a function of area for a given section shape). This introduces a nonlinear constraint coupling $A\_e$ and $L\_e`. Traditional algorithms handle this via iterative linearization or conservative approximations. A recent paper by Cai and Zhou (2024) explicitly incorporated code-based buckling strength (from a design code formula) into truss topology optimization[[20]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=of%20bars%20and%20residual%20stresses,the%20critical%20buckling%20strength%20is). They used a **linearized iterative approach**: at each iteration, they calculate an **allowable stress** for each member based on its current buckling capacity, then impose that as a linear constraint for the next iteration[[22]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=these%20problems%2C%20a%20novel%20topology,The). Essentially, the buckling constraint is turned into an adaptively updated stress limit (tighter for longer slender bars). This method successfully ensured no bar violates stability criteria at optimum and is more systematic than a fixed AASI threshold[[21]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=linearized%20iterative%20allowable%20stress%20method,the%20remarkable%20effect%20of%20including)[[23]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=proposed%20computational%20model%20shows%2C%20through,Topology%20optimization%3B%20Practical%20design%20code).
* **Global (System) Stability Constraints:** Local buckling checks may not be sufficient if the structure as a whole can buckle as a mechanism. Researchers like Rozvany (1996) pointed out that optimizing with only local buckling and stress constraints can lead to *erroneous (unstable) designs*[[24]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=Abstract). One remedy is to include **global stability constraints**, for instance requiring the lowest eigenvalue of the global stiffness matrix (or geometric stiffness for a buckling analysis) to be above a threshold. Such **eigenvalue constraints** turn the problem into a nonlinear eigenvalue optimization, which is challenging but can be tackled with iterative schemes. Rozvany suggested that adding system stability constraints (global buckling modes) or introducing slight imperfections in the model can avoid the singular, unstable solutions[[25]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=A%20serlous%20difficulty%20in%20topology,without%20changing%20the%20ground%20structure). In practice, some studies impose a global buckling load factor constraint (e.g., the structure must support some multiple of the load before global buckling). Solving this usually involves computing the smallest eigenvalue of a generalized eigenproblem $K \mathbf{u} = \lambda K\_{\text{geom}} \mathbf{u}$ each iteration, and using its sensitivity in optimization – a heavy but doable approach with modern solvers.
* **Sequential Linear/Convex Approximations:** Because buckling constraints are highly non-convex, sequential approximation methods (like the SCP framework itself or others like MMA) are commonly applied. For instance, **Zhou and Rozvany (1996)** and others used an optimality criteria or sequential linear programming approach where at each step the buckling constraint is linearized or turned into a simpler form[[24]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=Abstract). More recently, **Pedersen et al. (2018)** (as referenced in *Structural and Multidisciplinary Optimization*) solved stress- and local-buckling-constrained truss optimization via sequential **linear programming** with move-limits to ensure stability of the algorithm (this is conceptually similar to SCP, but using linear subproblems)[[26]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=Image). The use of conservative linearization ensures feasibility with respect to buckling is maintained as the design updates.
* **Design Codes and Safety Factors:** Another practical approach, if not focusing on theoretical optimality, is to incorporate design code formulas or safety factors for buckling directly into the optimization. The AASI threshold you implemented is one such idea – essentially a safety factor that if compressive force > X, then beef up the area. In the literature, a common simplification is to use different allowable stress for tension and compression (e.g. compression allowed stress = some fraction of yield stress)[[27]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=process%2C%20thus%20yielding%20suboptimal%20results,lower%20stress%20constraint%20for%20bars). This indirectly accounts for buckling by making compressive members thicker. While easy to implement, it can be overly conservative or not conservative enough, depending on the structure. The method of Cai and Zhou (2024) mentioned above is an improvement in that they compute the allowable compression stress from an **explicit buckling strength formula each iteration** rather than a fixed fraction[[22]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=these%20problems%2C%20a%20novel%20topology,The)[[28]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=process%2C%20thus%20yielding%20suboptimal%20results,lower%20stress%20constraint%20for%20bars). This might be something to consider: integrating a standard code’s buckling curve (like Eurocode or AISC formulas) and updating each member’s minimum required area accordingly each iteration.

In summary, to solve problem C you would likely want to **embed buckling constraints into the optimization loop more formally**. This could mean using a sequential approximation (treating buckling load as a constraint that is linearized each step, similar to how compliance is handled) or leveraging an existing algorithm like MMA that can natively handle such non-linear constraints robustly[[29]](https://orbi.uliege.be/bitstream/2268/100151/1/OA56.pdf#:~:text=3,MMA)[[30]](https://orbi.uliege.be/bitstream/2268/100151/1/OA56.pdf#:~:text=generalization%20of%20the%20pure%20Method,2%20Quadratic%20Separable%20Approximations). Keep in mind Rozvany’s warning that even with these constraints, certain ground-structure formulations can produce singular designs[[25]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=A%20serlous%20difficulty%20in%20topology,without%20changing%20the%20ground%20structure), so sometimes a slight modification (e.g. preset minimal connectivity or imperfection) is needed for a well-behaved solution.

## Conclusion

In tackling problems B, C, D, the literature suggests a combination of **improved algorithmic strategies** and **more integrated modeling** of the physics. For the oscillation and trust-region issues (B), modern SCP methods favor softening the trust region (through penalties or higher-order terms) and adaptively detecting oscillations[[5]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=oscillation%20phenomenon%20or%20solution%20chattering,trust%20region%20radius%20terms%20with)[[2]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=,optimality%20of%20the%20SCP%20method). Coupled with that, addressing the root cause – the design-dependent load – via smoothing or partial coupling (as in SAND or load interpolation approaches) will make the optimization more robust[[14]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=,Those)[[11]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=Solution%20procedures%20in%20structural%20optimization,In%20this%20paper%2C%20we%20make). For the buckling constraint issue (C), moving beyond a simple threshold to a proper constraint handled by sequential approximation or an optimizer like MMA will yield designs that are both optimal and safe[[22]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=these%20problems%2C%20a%20novel%20topology,The)[[24]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=Abstract). None of these improvements contradict the SCP framework; in fact, they enhance it: by making subproblem approximations more faithful (including load sensitivities, buckling behavior) and the algorithm more stable (less oscillation, no omitted failure modes). Implementing these techniques – drawn from recent research and classical methods – should directly help solve the current challenges (B, C, D) in your project. Each method comes with computational overhead (e.g. computing load sensitivities or eigenvalues), but given the critical nature of convergence and feasibility, they are worthwhile considerations to obtain a reliable and optimal solution.

**Sources:** Recent SCP convergence enhancements[[5]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=oscillation%20phenomenon%20or%20solution%20chattering,trust%20region%20radius%20terms%20with)[[7]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=); design-dependent load handling in optimization[[15]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=3D%20compliant%20mechanisms%20and%20loadbearing,a%20function%20of%20the%20design)[[14]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=,Those); simultaneous analysis-design strategies[[11]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=Solution%20procedures%20in%20structural%20optimization,In%20this%20paper%2C%20we%20make)[[13]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=conditioning%20inherent%20in%20penalty%20reformulations,are%20promising%20for%20future%20developments); buckling constraint formulations in truss optimization[[20]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=of%20bars%20and%20residual%20stresses,the%20critical%20buckling%20strength%20is)[[25]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=A%20serlous%20difficulty%20in%20topology,without%20changing%20the%20ground%20structure).

[[1]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=the%20trust%20region%20radius%20is,a%20soft%20constraint%20when%20oscillation) [[2]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=,optimality%20of%20the%20SCP%20method) [[5]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=oscillation%20phenomenon%20or%20solution%20chattering,trust%20region%20radius%20terms%20with) [[6]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=oscillation%20phenomenon%20or%20solution%20chattering,occurrence%20condition%20judgment%20is%20proposed) [[7]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=) [[8]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=function%20oscillation%20phenomenon%20occurs%2C%20the,the%20oscillation%20and%20improve%20convergence) [[9]](https://www.mdpi.com/2226-4310/11/9/785#:~:text=To%20overcome%20the%20oscillation%2C%20the,the%20direction%20with%20the%20minimum) Sequential Convex Programming for Reentry Trajectory Optimization Utilizing Modified hp-Adaptive Mesh Refinement and Variable Quadratic Penalty

<https://www.mdpi.com/2226-4310/11/9/785>

[[3]](https://www.researchgate.net/publication/358888469_Oscillation_Phenomenon_in_Trust-Region-based_Sequential_Convex_Programming_for_the_Nonlinear_Trajectory_Planning_Problem#:~:text=suitable%20for%20onboard%20aerospace%20applications,via%20a%20complex%20reentry%20trajectory) [[4]](https://www.researchgate.net/publication/358888469_Oscillation_Phenomenon_in_Trust-Region-based_Sequential_Convex_Programming_for_the_Nonlinear_Trajectory_Planning_Problem#:~:text=affecting%20the%20convergence,contribution%20of%20this%20paper) (PDF) Oscillation Phenomenon in Trust-Region-Based Sequential Convex Programming for the Nonlinear Trajectory Planning Problem

<https://www.researchgate.net/publication/358888469_Oscillation_Phenomenon_in_Trust-Region-based_Sequential_Convex_Programming_for_the_Nonlinear_Trajectory_Planning_Problem>

[[10]](https://depts.washington.edu/uwrainlab/wordpress/wp-content/uploads/2020/01/ACC_2020.pdf#:~:text=it%2C%20followed%20by%20a%20simple,SEQUENTIAL%20CONVEX%20PROGRAMMING) depts.washington.edu

<https://depts.washington.edu/uwrainlab/wordpress/wp-content/uploads/2020/01/ACC_2020.pdf>

[[11]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=Solution%20procedures%20in%20structural%20optimization,In%20this%20paper%2C%20we%20make) [[12]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=alternately%20in%20an%20iterative%20scheme,These%20schemes%20give%20rise%20to) [[13]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=conditioning%20inherent%20in%20penalty%20reformulations,are%20promising%20for%20future%20developments) [[16]](https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1#:~:text=An%20advantage%20of%20the%20simultaneous,first%20is%20a%20projected%20Newton) Simultaneous structural analysis and design based on augmented Lagrangian duality | Structural and Multidisciplinary Optimization

<https://link.springer.com/article/10.1007/BF01742636?error=cookies_not_supported&code=125ad5d3-5f30-47aa-b3c6-ef9dd830cfc1>

[[14]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=,Those) [[15]](https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf#:~:text=3D%20compliant%20mechanisms%20and%20loadbearing,a%20function%20of%20the%20design) orbit.dtu.dk

<https://orbit.dtu.dk/files/238101658/marac_nme.6618.pdf>

[[17]](https://onlinelibrary.wiley.com/doi/full/10.1002/eng2.12514#:~:text=On%20stability%20and%20relaxation%20techniques,with%20the%20hydrodynamic%20condition) On stability and relaxation techniques for partitioned fluid‐structure ...

<https://onlinelibrary.wiley.com/doi/full/10.1002/eng2.12514>

[[18]](https://www.researchgate.net/publication/327139340_Iterative_coupling_algorithms_for_large_multidomain_problems_with_the_boundary_element_method#:~:text=,the%20boundary%20element%20method) (PDF) Iterative coupling algorithms for large multidomain problems ...

<https://www.researchgate.net/publication/327139340_Iterative_coupling_algorithms_for_large_multidomain_problems_with_the_boundary_element_method>

[[19]](https://ui.adsabs.harvard.edu/abs/2018IJNME.115..825E#:~:text=Level%20set%20topology%20optimization%20for,to%20solve%20the%20compliance) Level set topology optimization for design-dependent pressure load ...

<https://ui.adsabs.harvard.edu/abs/2018IJNME.115..825E>

[[20]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=of%20bars%20and%20residual%20stresses,the%20critical%20buckling%20strength%20is) [[21]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=linearized%20iterative%20allowable%20stress%20method,the%20remarkable%20effect%20of%20including) [[22]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=these%20problems%2C%20a%20novel%20topology,The) [[23]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=proposed%20computational%20model%20shows%2C%20through,Topology%20optimization%3B%20Practical%20design%20code) [[27]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=process%2C%20thus%20yielding%20suboptimal%20results,lower%20stress%20constraint%20for%20bars) [[28]](https://app.iass2024.org/files/IASS_2024_Paper_421.pdf#:~:text=process%2C%20thus%20yielding%20suboptimal%20results,lower%20stress%20constraint%20for%20bars) A

<https://app.iass2024.org/files/IASS_2024_Paper_421.pdf>

[[24]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=Abstract) [[25]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=A%20serlous%20difficulty%20in%20topology,without%20changing%20the%20ground%20structure) [[26]](https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d#:~:text=Image) Difficulties in truss topology optimization with stress, local buckling and system stability constraints | Structural and Multidisciplinary Optimization

<https://link.springer.com/article/10.1007/BF01197036?error=cookies_not_supported&code=5e12a964-51e0-46c7-aa98-7c45d67d178d>

[[29]](https://orbi.uliege.be/bitstream/2268/100151/1/OA56.pdf#:~:text=3,MMA) [[30]](https://orbi.uliege.be/bitstream/2268/100151/1/OA56.pdf#:~:text=generalization%20of%20the%20pure%20Method,2%20Quadratic%20Separable%20Approximations) dcamm.dvi

<https://orbi.uliege.be/bitstream/2268/100151/1/OA56.pdf>