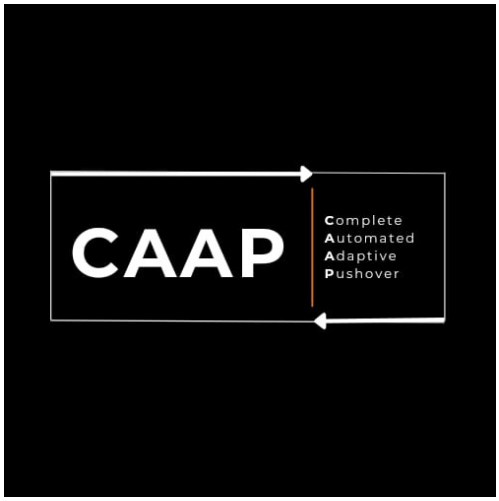
CAAP

(***C**omplete **A**utomated **A**daptive **P**ushover*)



Users Manual

Developers: Maximilian Osterkamp and Dominik Matthias Kämper

Computational Applied Mechanics – University of Wuppertal,
March 2024

The following documentation focuses on the use of the (Matlab) "CAAP" tool together with SAP2000 to perform automated adaptive pushover analyses on FE models with truss and beam elements.

I. General information

This program enables a semi- or fully automated adaptive pushover calculation with SAP2000. The following calculation methods are available with the corresponding options (regarding adaptivity, full automation and load distribution):

- ‚Standard' Pushover procedure
  - non-adaptive/adaptive
  - if adaptive: Semi- or fully automated
  - Load pattern: mass, mode as well as mass & mode proportional
  - with/without subsequent seismic design check (e.g. determination of performance point)

  No adaptive calculation is possible with a purely mass-proportional load distribution. Apart from this, all other conceivable combinations can be selected.
- Modified AMI procedure with constant spectral acceleration increments $\Delta S_{a,B}^{(i)}$
  - only fully automated adaptive (includes seismic design check in each case)
- Modified AMI method with optimized (varying) reference spectral acceleration increments
  - only fully automated adaptive (includes seismic design check in each case)

Both the adaptive implementation of the "standard" pushover calculation and the modified AMI method with optimized spectral acceleration increments are based on an analysis of the (dimensional) pushover curve in each adaption step and a resulting adjustment of the calculation steps (see. Chapter III).

II. General program flow

The following Figure 1 shows the general program flow of the CAAP tool, but is only intended to provide a rough overview of the general calculation steps. Details of the application-relevant steps are explained in the following Chapter III.

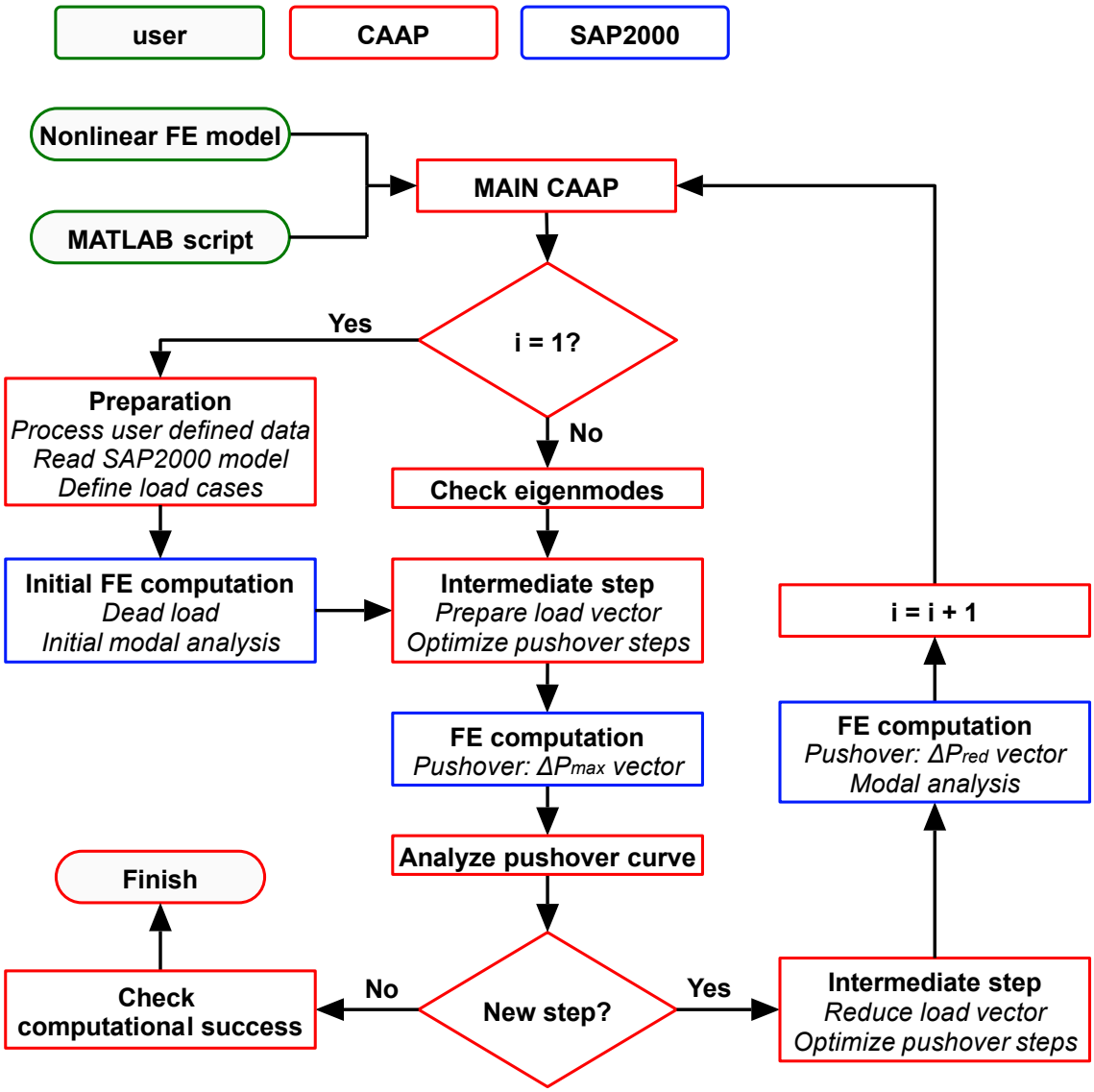


Figure 1: Flow chart - Application of the CAAP algorithm

III. User-side specifications

The user must essentially (or, in the case of a fully automated calculation, exclusively) perform two steps:

1. Creation of a numerical model and preliminary investigation in SAP2000
2. Creation of a calculation script in Matlab

These two steps are explained in more detail below.

1. Creation of a numerical model and preliminary investigation in SAP2000

First, the structure must be modeled completely, including the potential yield points. Three load cases must already be created: a load case for the permanent vertical loads due to self-weight (nonlinear static load case), a load case for the modal analysis based on the stiffness distribution at the end of this load case and a nonlinear static pushover load case linked to the stiffness distribution and load as well as structural responses at the end of the vertical load case.

For the latter, in the case of a "standard" pushover calculation, the "Displacement controlled" option and "Use monitored displacement" and a sufficiently large target displacement of the control node must be selected. If the modified AMI method is used, however, the "Full Load" option must be selected. Here, the primary earthquake direction is to be selected for the direction of the monitoring point displacement. Generally, a node that has relatively large deformations in the primary earthquake direction in the basic mode shape(s) should be selected as the control node. However, the latter can only be assessed more precisely after an initial manual modal analysis and may therefore need to be adjusted accordingly.

An initial manual modal analysis can then be carried out based on the system properties at the end of the nonlinear calculation for the permanent vertical loads. Based on the results of this preliminary modal analysis, a first impression can already be gained,

a) which natural modes should be considered in which earthquake directions (assessment using the effective modal masses). The mode shapes in SAP2000 must be used to assess the sign with which this should be done, i.e. whether the load distribution should act in the direction of the respective natural mode or in the opposite direction. In most cases, different modal load combinations with varying signs of the modes and with different reference modes will be required. *This point a) is only irrelevant for a purely mass proportional load distribution in the case of a standard pushover calculation.*

b) which node is suitable as a monitoring point (*mp*) for the applied primary earthquake direction and there for the relevant modes (*ideally a node with a large relative deformation component in the respective eigenvectors, at least no node WITHOUT deformation in the considered direction for the considered modes*).

2. Creation of a calculation script in Matlab

For each calculation with the "CAAP-Tool", a calculation script must be created in which the essential specifications are made, followed by the function call of the main routine "MAIN_CAAP". The "specifications" mentioned are briefly explained below in relation to the procedure.

Specification of general "information" in subfields of „arg.info“ (‚information‘):

- *sap_path* (required, string): Path of SAP2000.exe
- *sap_file* (required, string): Path of FE model ($2k file)
- *export_file_name* (optional, string): Name of automated exportfile (xml), default: ‚Auto_export‘
- *name_vert* (required, string): Name of the nonlinear static load case for the gravitational loads
- *name_modal* (required, string): Name of the load case for the initial modal analysis
- *name_pushover* (required, string): Name of the load case for the initial nonlinear static pushover analysis
- *procedure* (optional, string): Specify which procedure is to be used:
  - ‚standard‘ (default, „standard“ Pushover calc.)
  - ‚ami_c‘ (modified AMI with **c**onstant $\Delta S_{a_B}^{(i)}$)
  - ‚ami_o‘ (modified AMI with **o**ptimised $\Delta S_{a_B}^{(i)}$)
- *console* (optional, logical/int [0 | 1]): Are detailed console outputs desired by the user?
- *sound* (optional, int [0 | 0,5 | 1]): Is an acoustic signal desired at the end of a calculation (concerning calculation success/failure) by the user?
  - 0: No
  - 0.5: Yes, but very briefly („horn“ for success)
  - 1: Yes („Steigerlied“ for success, default)
- *mail* (optional): Do you want an automated e-mail notification in the event of a required user input?

  If so:
  Define substructure with fields:
  - „mailadress“, e.g. ‚doe@uni-wuppertal.de‘
  - „name“, e.g. ‚doe‘
  - „password“, e.g. ‚Password123‘
  - „smtp_server“, e.g. ‚mail.uni-wuppertal.de‘

  (→ „subject“ and „content“ are generated automatically by ‚caap_check_varargin‘ or in the respective function that has identified the requirement of the user input)

Specification of calculation parameters in subfields of „arg.comp“ (‚computation‘):

- *nl_steps* (optional, [1x2] array): Number of ‚Multiple States‘ of nonlinear Pushover analyses, as: [min max], default [10 100]
  Data deviating from the default must be defined in Matlab!
- *d_tol* (optional, float): If no convergence was reached and even tripling (!!!) of the "Total Steps" was unsuccessful:
  Adjust (increase) the error tolerance by Δ_{tol};
  Value 0 means: no adjustment, default: 0
- *adaptive* (optional, logical/int [0 | 1]): Should the calculation be adaptive?
  default:
  - case „standard“: *false* or 0
  - case „ami_c“ & „ami_o“: *true* or 1, whereby the specification of a non-adaptive calculation is even intercepted with a corresponding warning!
- *k_loc* (optional, float [$0 \leq k\_loc \leq 1$]): Local ratio value between the current slope and the slope of the previous point (kink point); only relevant in the case of „standard“ or „ami_o“, default: 0.9
- *k_glob* (optional, float [$0 \leq k\_glob \leq 1$]): Global ratio value between the current slope and the slope at the start of the current segment; only relevant in the case of „standard“ or „ami_o“, default: 0.9
- *delta_s_a_b* (optional, float): (for ‚ami_c‘ and also senseful for ‚ami_o‘, if negative spectral increments occur later and the procedure thus is changed to ‚ami_c‘ from then on) Spectral acceleration increment of the reference mode (constant value for all adaption steps), default: 0.01 [m/s^2]
- *auto_run* (optional, logical/int [0 | 1]): Should the calculation be fully automatic?
  default:
  - case „standard“: *false* or 0
  - case „ami_c“ & „ami_o“: *true* or 1 (is set automatically)
- *algodec* (optional, logical/int [0 | 1]): (optional bei ‚ami_k‘/‚ami_o‘) Should the algorithm be given more decision-making power in terms of potential mode-self-assignments?
  default: *false* or 0
- *check* (optional, logical/int [0 | 1]): Should a seismic design check be carried out? The question only really arises in the „standard“ case; „ami_c“ and „ami_o“ alway include a seismic design
  default:
  - case „standard“: *false* or 0
  - case „ami_c“ & „ami_o“: *true* or 1, (is set automatically)

- *load_pattern* (optional, string): Specifies which proportionality the load distribution should have; only relevant in the „standard“ case:
  - ‚mass‘ (default)
  - ‚modal‘
  - ‚mass_modal‘
- *push_load_ref* (optional, string): Specification of the reference for the nodal loads
  - ‚frames‘ (default): all (system & FE) nodes are considered; pure ‚frame‘ model required
  - ‚joints‘: all system nodes are considered (no automatic meshing does take place in this case); *all* elemttypes are possible (also shell) ATTENTION*: Here, the local coordinate systems of all (not completely restrained and thus) loaded nodes have to correspond to the global cs (X-Y-Z)!*
- *d_earthquake* (required, string): Earthquake directions to be taken into account, defined as ‚YXZ‘; first letter: primary seismic direction
- *modes* (possibly required*, struct): (*except in case of „standard“, Load pattern „mass“ & check = 0) Eigenmodes to be considered in specific directions, Specification for each direction of *d_earthquake* definition as: modes.Y = [1 3 -4], modes.X = [-4]
  - case „standard“: Only one mode may be specified in each earthquake direction. In the case of mode shape or mass & mode shape proportional load distribution, the sign determines whether the loads act in the direction of the respective mode shape or in exactly the opposite direction.
  - case „ami_c“ & „ami_o“: The signs correspond to the α factors by Norda. A mode may theoretically be applied in all directions.
- *modes_dir_protected* (optional, array [1x$N_{modes,protected}$]) Which modes are to be protected in terms of direction (whereby *„SF“ is set normally*)?
- *sign_factors* (only required in case of „standard“ & „mass“) Information to be defined in relation to the direction of earthquakeas to whether the respective load distribution should act in a positive or negative global coordinate direction, -1 or 1 as [1 x $n_{directions}$] array
- *dir_factor* (optional, [1xN] array): Superposition factors for a multidirectional modified AMI calculation, default: [1.0 0.3 0.3]
- *xi_0* (optional, float): Basic damping in the event of an earthquake [%], default: 5%
- *hb* (possibly required*, string): (* wenn *nw* = 1) Hysteretic behavior (acc. to ATC40 (1996)): ‚A‘, ‚B‘ or ‚C‘

<u>Specification of the elastic response spectrum in subfields of „arg.rs“ (‚response spectrum‘):</u>
Relevant in the case of a „standard“ pushover calculation with a subsequent seismic design check or a calculation according to the modified AMI method

The following specifications are possible or generally to be made:

- *standard* (optional, String): Normative standard for the definition of the rs, currently available: „ec8-1“ (default) & „ec8-1_de“ (G.N.A)
- *a_g* (ggf. required, float): Horizontal design ground acceleration a_g [m/s²]; required for seismic excitations in ‚X‘ & ‚Y‘ direction
- *a_vg* (ggf. required, float): Vertikal design ground acceleration a_{vg} [m/s²]; required for seismic excitation in ‚Z‘ direction
- *S* (required für hor. rs, float): Soil parameter (depending on the soil class and the response spectrum type)
- *T_BCD* (required, [1x3]-Array): Control periods T_B, T_C, and T_D [s] (as floats depending on the soil class & the response spectrum type)
- *dT* (optional, float): Resolution of the time axis [s], default: 0.01 s
- *T_max* (optional, float): Upper limit value of the time axis, default: 4 s
- *T_min* (optional, float): Lower limit value of the time axis, default: 0 s

Concluding remarks:
The direction ('horizontal' or 'vertical') of the response spectrum is determined automatically (i.e. not directly by the user) and saved under the arg.rs.direction field. In the case of a „standard“ pushover calculation, this is done immediately before the „caap_CSM_determine_pp“ routine is called and in the case of a calculation using the modified AMI method within the „caap_pushover_pointloads“ function separately for the respective directions of seismic excitation.
The damping correction factor η is determined independently within the „caap_el_accel_response_spectrum“ routine. In the case of a „standard“ pushover calculation, this results from the current effective total damping $\xi_{eff}^{(i)}$, whereas in the case of a calculation using the modified AMI method, it results from the respective effective modal damping $\xi_{n,eff}^{(i)}$:

$$\eta = max\left\{\sqrt{10/\left(5+\xi_{(n,)eff}^{(i)}\right)},\ 0.55\right\}. \qquad (1)$$

3. Further application-relevant details

Possible changes to the numbers of modified modes in the course of an adaptive calculation:
One challenge with automated adaptive pushover calculations is that slightly different eigenmodes can change their number between two adaption steps.
In the CAAP tool presented here, this problem was taken into account by the fact that the algorithm checks precisely this in the following two ways at the beginning of each adaption step:

1. First, it is checked whether the mode *n* can be clearly assigned to a new mode *m* of the current, *i*-th modal analysis with $M_{eff,m,R}^{(i)}$ with regard to the effective modal masses $M_{eff,n,R}^{(i-1)}$ in all directions *R* (with $M_{eff,n,R}^{(i-1)} > 2\%$) relevant for this mode. Here, *clearly* means that in a corridor of up to four neighboring modes (e.g. for reference mode 2: modes 1, 2, 3, 4, 5, 6)...
   - there is a *new* effective modal mass $M_{eff,m,R}^{(i)}$ for each relevant direction *R*, for which the difference between $M_{eff,m,R}^{(i)}$ and $M_{eff,n,R}^{(i-1)}$ (*Dif 1*) in direction *R* is at most half the difference to the second closest effective modal mass (*Dif 2*), but the latter in turn is at least 4%: *Dif 2* ≥ max{2 - *Dif 1*; 4%}
   - all analyzed directions *R* provide the same number *m*
   - if $M_{eff,n,R}^{(i-1)}$ is assigned to $M_{eff,m,R}^{(i)}$, the effective modal mass $M_{eff,m,R}^{(i-1)}$ is not also most similar to $M_{eff,n,R}^{(i-1)}$
   - the following applies to the ratio of eigenperiods: $max\{T_n^{(i-1)}, T_m^{(i)}\}/min\{T_n^{(i-1)}, T_m^{(i)}\} \leq 1.3$

Note: A „1:1“ mode interchange based on the effective modal masses of a specific direction must be confirmed by the user, as these are often incorrect.

2. *If the mode assignment via the effective modal masses fails,* the mode shape from the previous modal analysis is compared to the current mode with the same number and, if the deviation is correspondingly large, also with its four neighbors „on both sides“ for each mode considered. Specifically, a „comparison“ of two modes includes the mean difference in the node displacements (*sum of the node displacement differences divided by the number of nodes*), which is determined separately for all three global directions. Here, the „old“ reference mode is given a specific sign and the respective current comparison modes are applied in both directions (i.e. + & -). All modes (*reference and comparison modes*) are normalized to the largest translation across all directions. The fact that a (current comparison) mode now corresponds to the modification of the respective previous reference mode under consideration is identified on the basis of the smallest difference sum. However, it is important to mention that these differences are only considered with regard to those directions in which both modes have „significant“ deformations (*> 0.01 after normalization with a maximum displacement at a node in a direction of exactly +1.0*).

The comparison with the respective predecessor (from step (*i-1*)), instead of the initial form of the respective mode, as well as the upstream check as to whether this mode has changed noticeably at all (identification by a mean limit difference – diff. of the normalized modes $n_I^{(i-1)}$ and $n_I^{(i)}$, divided by the number of nodes – of 5%), has two major advantages:

1. A successively changing mode can be assigned much more easily (or possibly at all) between step (*i-1*) and (*i*) than in a comparison with the mode from the initial modal analysis.
2. Checking whether a more precise „change“ consideration is worthwhile, taking into account the up to 8 neighbors, means that in the case of conflicts that require user input (more on this later), this does not have to take place from the first occurrence in every step.

If, for example, two modes are clearly swapped, two rows are created in an [n x 3] matrix, whereby this matrix is structured as follows:

$$arg.comp.modes.changes = \begin{bmatrix} n_I^{(1)} & n_I^{(i-1)} & n_I^{(i)} \\ n_{II}^{(1)} & n_{II}^{(i-1)} & n_{II}^{(i)} \\ \vdots & \vdots & \vdots \end{bmatrix}. \quad (2)$$

Here, $n_I^{(1)}$ describes the number of the I. Considered mode (only these are examined) being affected by a swap according to the initial modal analysis (in step 0 or 1), $n_I^{(i-1)}$ the number from the previous modal analysis in step (*i-1*) and $n_I^{(i)}$ the new number in the current *i*-th step. The background to this is that the modal capacity spectra and all associated variables (such as the effective yield points, effective natural frequencies, etc.) are always saved under the initial number. The last point in the respective capacity spectrum is therefore accessed via the value in the first column of *arg.comp.modes.changes*. The current modal results from SAP2000, such as participation factors, current natural frequencies, etc., on the other hand, are accessed via the current mode number. The current link then provides the additional third column of *arg.comp.modes.changes*. This memory structure also allows for the case that a mode changes back or even receives a new number. In this case, only the second and third columns need to be adapted. *In the meantime, each mode change is checked to see whether the displacement signs of „significantly skewed modes“ ($|v|_{max,R2} > 0.5\text{-}|v|_{max,R1}$) match in both directions with the largest deformations. For this purpose, it is checked whether most of the same signs are present in each direction with mirrored or original reassigned mode. If equal: plausible (as in the case: only one direction with relevant deformations).*

Furthermore, each identified mode change is checked to see whether the newly assigned mode has a negative eigenperiod and is therefore to be classified as a „kinematic eigenmode“. If this is the case, the user is informed and the change is discarded, i.e. in case of doubt, the user must then manually assign a new mode to the old one.

If the assignment of modes is unclear, a warning is issued and the program sequence is not continued until the user has made a corresponding assignment (*after examining the relevant modes in SAP2000*) by means of a console input. This is the case, for example, if:

- two different modes $n^{(i-1)}$ of the previous modal analysis are assigned to the same current mode $n^{(i)}$ (i.e. $n^{(i)}$ occurs twice in the 3rd column of *arg.comp.modes.changes*),
- the separate, direction-related differences (when considering a certain mode $n^{(i-1)}$ with its current counterpart $n^{(i)}$ and its neighbors) show their minima at different modes and thus do not allow a clear assignment, or
- the final plausibility check in step *i* fails because a current mode $n^{(i)}$ (e.g. 2), which has already been considered, has been assigned to another previous mode $n^{(i-1)}$ (e.g. 1) without a new current mode being assigned to this mode (here: 2).

In the latter case, the user is also given the opportunity (via a corresponding query) to enter further changes that he or she has identified by looking at the current and previous modes independently, until the algorithm continues. If the algorithm has associated an initial mode with a change that does not actually change its number, the user can enter the same number and the algorithm deletes the corresponding meaningless line from *arg.comp.modes.changes* later.

"Ultima Ratio" check with regard to a potential „self-assignment“ of a specific mode:
Before the prompt if the assignment of a mode is unclear, there is an ultima ratio check that attempts to prove a self-assignment of the mode. This is done in the following two stages:

- *a general verification stage, which is quite safe and is always carried out*
- *a second stage with somewhat „less certain criteria“, which could therefore lead to an incorrect self-assignment in exceptional cases; this is only carried out if the user allows the algorithm to do so via „arg.ber.algodec = 1 or true“ („algorithm decisions“) – if this secondary checking stage was necessary, the user is at least informed of this, so that he or she should examine the mode changes in this step even more critically than usual and then repeat the calculation with „arg.ber.algodec = 0 or false“ in the aforementioned case of an incorrect self-assignment and then make a user input at this point.*

Concrete reviews of the two levels mentioned above:

- *The first stage consists of two examinations A) and B):*
  - A) *Is the previous (old) eigenperiod of the current mode most similar to the new eigenperiod of the same mode AND is the difference between these two periods not larger than 50% of the next largest difference?*
  - B) *For the „versions“ $\vec{\Phi}_n^{(i-1)}$ and $\vec{\Phi}_n^{(i)}$ of the n-th mode, is the direction R_1 of the largest deformation and the direction R_2 of the second largest deformation identical AND is the ratio of $|v_{R_1}|_{max}$ to $|v_{R_2}|_{max}$ greater than 2 in each case?*
- *The second stage is a sub-item of the higher-level check A) and therefore attempts to „justify“ a self-assignment with regard to frequency development even if the above-mentioned criterion A) is not fulfilled. There are three possibilities here:*
  1. *Proof that the above-mentioned „most similar frequency“ rule refers back to the current mode, even if the 50% criterion of the difference ratio does not apply.*
  2. *Proof when assigning to a new mode number that the "opposite" of $T_{mode,current\ no.}^{(i-1)}$ or $T_{mode,new\ no.}^{(i)}$ cannot be assigned to any other partner (e.g.: $T_1^{(i-1)}$ fits best to $T_2^{(i)}$, but $T_1^{(i)}$ also fits best to $T_1^{(i-1)}$ OR $T_2^{(i-1)}$ also fits best to $T_2^{(i)}$).*
  3. *Proof that all (maximum 3) periods „around the current mode number“, i.e. for mode 2: T_1, T_2 and T_3, have all increased to a similar extent.*

Overall, a self-assignment is only considered „justified“ if A) AND B) are fulfilled.

Signs of individual (directional and mostly modal) load distributions:
As already explained before, the signs or directions of the load distributions for the different methods and load approaches can be specified as follows:

- „Standard“ Pushover procedure:
  - Mass proportional load approach:
    arg.ber.sign_factors = [-1 1], (for two directions of seismic excitation considered)
  - mode or mass and mode proportional load approach:
    arg.comp.modes.Y = [-1], arg.comp.modes.X = [3] (mode 1 negative, mode 3 positive)
- Calculation according to the AMI method („ami_c“ or „ami_o“):
  arg.comp.modes.Y = [-1 5 12], arg.comp.modes.X = [3 -6 -1] (i.e. $\alpha_1^{(k)}=-1$, $\alpha_3^{(k)}=1 \ldots$ *)*

Caution: In the AMI procedure, a certain mode may be applied in several directions, but not with different signs!
Overall, the signs of the individual pre-factors of the load vector can be deliberately eliminated by a corresponding magnitude function due to the points mentioned above, which is also done!

Against this background, the CAAP tool always uses the amount of the participation factor $\beta_{n(R)}^{(i)}$, since the components of the mass matrix are positive anyway as well as the spectral acceleration increment in the mod. AMI method, so that in this way a modal load distribution ...

- with a positive sign factor acts in the direction of the corresponding mode shape
  (*or in the case of the „standard“ pushover method: in positive global coordinate direction*)
- and in the opposite direction if the sign factor is negative.

Furthermore, in later adaptation steps, the tool ensures that the „direction“ of the eigenmodes randomly determined by the eigenvalue solver in SAP2000 remains the same in relation to the initial analysis by setting a scaling factor *SF* to -1 in the event of a change in the direction. Such a change of direction is also identified for constantly changing modal shapes using the sign of the node displacement in the respective direction (in case of doubt the „most important“ direction, i.e. „previously named“ in *arg.comp.modes.*R_i), which had the largest amount in the last step (*i-1*).

Later in this document, it will be explained in more detail that, due to changes in modes, initially pure „X eigenmodes“ may have to be also considered in the Y direction if these „develop“ Y deformation components in the course of the calculation. In such cases, it is not useful or possible to determine the sign based on the largest node displacement in the Y direction (if the deformations in the Y direction are all still approximately 0). The tool notices this in the subroutine sub_get_direction of the caap_check_eigenmodes function, which, as in the very first step (arg.info.number == 1), means that no check of a sign change takes place here and instead the calculation is performed with arg.comp.modes.SF(arg.info.number,i_mode) = 1.

Checking the effective modal masses of modes not taken into account:
Since the user determines at the very beginning, on the basis of an initial (independent) modal analysis, which modes he or she wants to consider in which earthquake direction, it may occur that in the course of the formation of plastic zones in the structure, certain previously unconsidered modes „develop“ an effective modal mass higher than 5%. This is exactly what the tool checks at the end of an AMI step in the routine *caap_analyze_modal_participating_mass_ratios*. A newly relevant mode is stored in an array with the step number of the first „becoming relevant“. If an unconsidered mode *n* becomes relevant for the first time in step *i*, a corresponding output is generated so that the user immediately has the option of canceling the calculation and performing a new AMI calculation with consideration of this mode from the start. However, it is important that the user looks at the mode *n* in step *i* and checks whether this also corresponds to the mode *n* in step 1 or – if this is not the case – which mode it corresponds to in the initial state. This must of course be specified for a new calculation.
However, an automatic consideration from the respective calculation step where the effective modal mass exceeds the limit value explained above is not possible, as this mode would then have no modal „capacity spectrum history“ and „starting from zero“ would then have a disproportionately high spectral increment compared to the other modes in the next step.

Identification of a strong change in certain (albeit assignable) modes:
The previously described "mode-change" identification is based on the investigation of „relevant“ deformations of the „old“ and „new“ modes. In concrete terms, this means that only directions are examined with regard to which significant deformations occur both in state *i-1* and in state *i* for a specific mode or its comparison modes.
As a result, it may occur that a certain mode *n* in states *i-1* and *i* has a relatively similar deflection line in the Y direction and is therefore assigned to itself, or changes its number, however still allows a clear „change assignment“, even though it has no or very small X or Z deformation components in state *i-1* and significantly larger ones in state *i*. In such a case, the mode assignment is certainly justifiable because it really is a modification of the same mode. Nevertheless, such a process should always be assessed by the user (at least retrospectively).

Before outlining how the algorithm checks such mode changes without assignment problems (in the course of the „mode change“ investigation), the background to this problem with regard to the urgent assessment recommendation in the application case should be illustrated that has occurred within the scope of a case study. The figures below show the strong change in modes 4 and 5 of the FH model of a reinforced concrete bridge in the course of the start of yielding. It can be seen there that due to the massive failure of some areas in the concrete as a result of tension, not only the bending stiffness but also the tensile stiffness of the piers decreases so considerably in some cases that the originally purely horizontal mode 5 increasingly shows vertical deformation components and the initially purely vertical mode 4 also shows significant horizontal displacements. Overall, it can be said that the completely different mode shapes 4 and 5 in the elastic system continue to converge, so that in an AMI calculation both would have to be taken into account from the outset in order to have the later also relevant mode shape 4 „on board“.

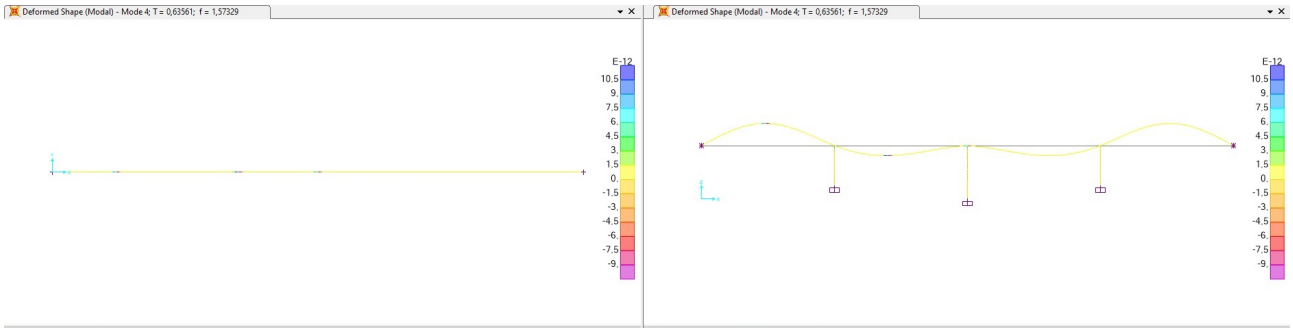


Figure 2: Elastic system - mode 4

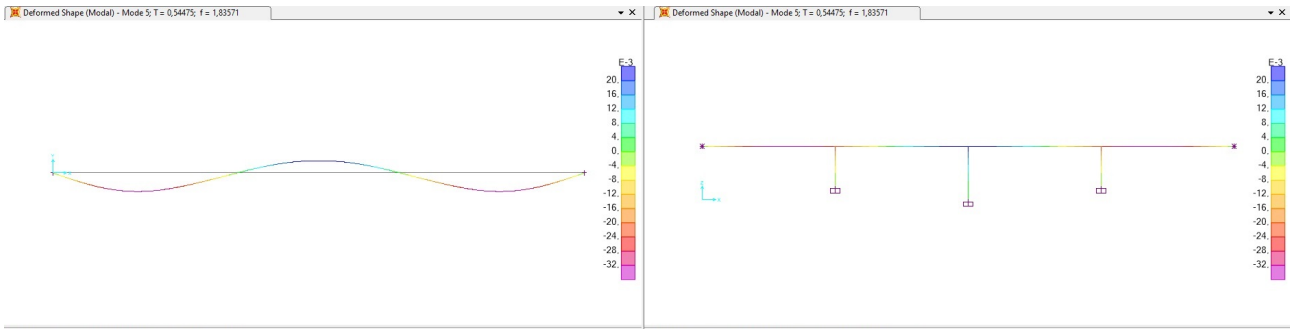


Abbildung 3: Elastic system - mode 5

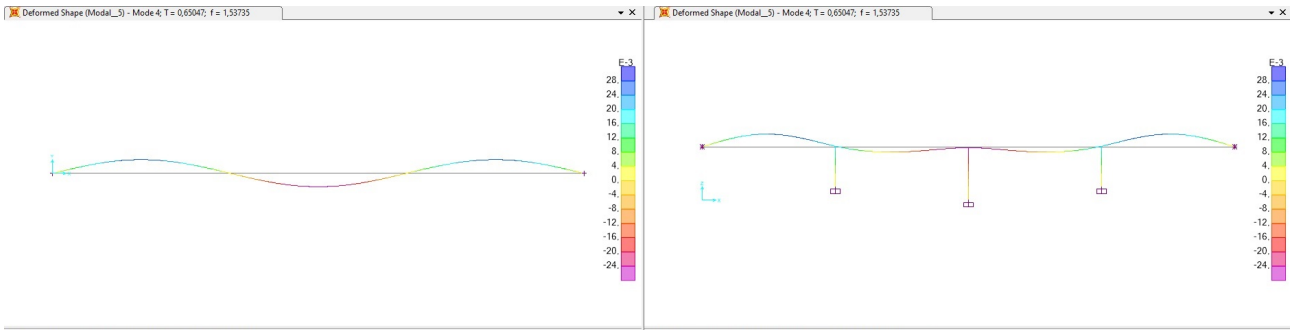


Abbildung 4: Inelastic system - mode 4

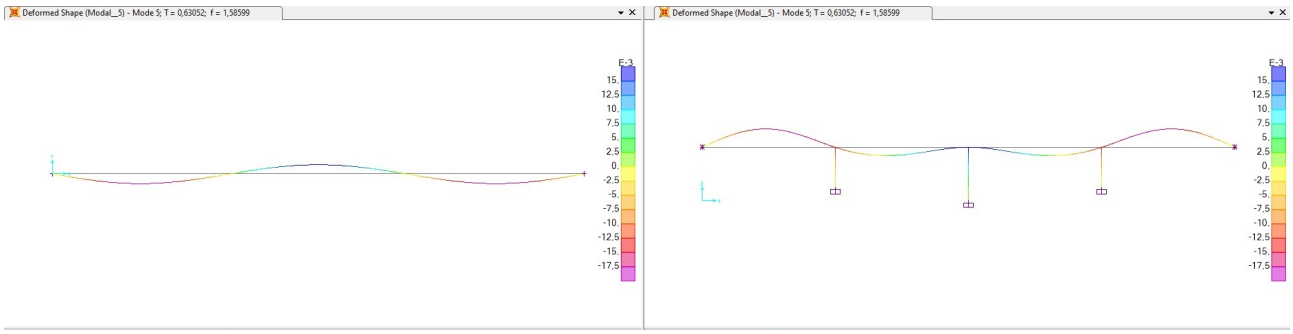


Abbildung 5: Inelastic system - mode 5

This check takes place within the routine *caap_check_eigenmodes,* and there in the subroutine *sub_check_mode_change*. Specifically, it checks whether a mode in its initial form has deformations of less than 1% of the maximum deformation across all directions in one direction and then more than 20% in this direction in the current form *i* (taking into account any „mode changes“). In such a case, however, there is only an informative output, as an interruption of the algorithm would certainly occur frequently and would then slow down numerous calculations.

Dealing with failed calculations:
Basically, the CAAP tool checks ...

- at the end of each calculation step of the „ami_c“ method or
- at the end of the correction step of an adaptive „standard“ or „ami_o“ calculation or
- following a „Max computation“ as part of an adaptive „standard“/„ami_o“ calculation, if *caap_analyze_pushover* does not consider a further step necessary, in case of „ami_o“ with $\Delta S_{a,B,max}^{(i)} < 0.005 \cdot max(\Delta S_{a,B,max}^{(1\ -\ i)})$, or no pushover curve analysis is possible in the aforementioned routine because there are fewer than two points for the current segment*,

whether the calculation was successful or not by reading out the LOG file. This specifically looks for „Error“ messages and the warning that the „Zero“ or „Total Steps“ have been reached. Other warnings are ignored.

** A check as to whether the calculation was successful only does not take place if the following „special case 1“, actually relating to a correction calculation, in this case the corresponding „Max computation“, is present.*

Special case 1:
The only exception here is the case of a „0“ calculation, which occurs during a correction calculation of the „ami_o“ method if it is determined during the determination of the new maximum spectral acceleration of the reference mode that the last existing spectral value roughly corresponds to this. In such a case, as already mentioned, a „0“ calculation is performed because this is algorithmically simpler than forcing an instantaneous calculation stop in the middle of the „analyze_pushover“ routine. However, such a calculation in SAP2000 always leads to zero steps being reached, even though the AMI calculation was still successful overall and does not have to be regarded as having failed. Consequently, it is precisely this special case that is intercepted in the *caap_check_calc_success* routine.

Special case 2:
Furthermore, in case of the AMI method with optimized spectral acceleration increments, it can happen that a „Max computation“ with the desired minimum number of pushover steps per load case („MinNumState“) works without any problems, but the subsequent correction calculation does not. Obviously, this cannot be a „genuine mechanical“ failure, as the structure was already able to absorb significantly greater loads. Rather, the reason for the failure is due to extremely small load increments (in such cases) in the correction calculation. *It may for example occur that the maximum incremental nodal load is around 0.1 kN and 20 pushover steps are performed. However, these extremely small load increments lead to equilibrium iteration problems at a load level of around 100 kN, so that a reduction of the pushover steps to 1 or 2 represents a solution to the problem.*

For this reason, the *caap_check_calc_success* routine checks whether the case described above exists before tripling the zero steps and possibly subsequently increasing the error tolerance (*by checking whether the maximum nodal load of the current, failed load case is less than 1% of the previous maximum load*). If this is the case, the user is asked to specify a factor *n* with which both the MinNumState and the MaxNumState are reduced equally. *For example: If the user enters a 10, instead of min. 20 and max. 100 steps, only min. 2 and max. 10 steps would be performed. When evaluating the input, the „floor“ function is used so that, for example, a minimum of 2 would be set for a mathematically reduced MinNumState of 2.1.*

If neither of the two special cases above is present, an attempt is made to „generate“ a successful calculation by...

1. Tripling the permissible „Total Steps“ and setting the permissible „Zero Steps“ to the same value (exactly once!)
2. Increase the iteration error tolerance by the value arg.comp.d_tol (maximum 4 increases!)
3. as „ultima Ratio“: set the nonlinear steps to 2 (MinNumState) or 5 (MaxNumState).

Only if all these steps also do not result in a successful calculation the calculation as a whole is considered to have failed (*arg.info.success = 0*).

Nonlinear Steps (defined as "Min Num State" & "Max Num State") of nonlinear calculations:
According to Chapter 2 of this document, the user can specify the number of nonlinear steps via the „nl_steps“ structure field of the „arg.comp“ structure. The use or deliberate avoidance of this desired minimum and maximum number of steps is explained in the following list.

The following specifications are set in the CAAP tool:

- The vertical loads are always calculated in 5 steps.
- In case of the "ami_c" procedure, all AMI steps are performed with the pushover step numbers defined in *arg.comp.nl_steps*.
- In case of the "ami_o" method, only the very first "Max computation“ is carried out with the number of pushover steps defined in *arg.comp.nl_steps*, as the most steps are required here (due to the greatest length of the total pushover path traversed) in order to detect the first kink point after the linear-elastic initial range accurately enough. All further "Max computations“ are then carried out with an optimized, lower number of steps in order to take into account the following two aspects that are very important for the performance of the method:
  1. A lot of computing time should not be „wasted“ unnecessarily in SAP2000 for calculations that have to „cope“ with a much smaller pushover path and can therefore also be mapped accurately enough with significantly fewer steps.
  2. A too high a resolution of the ever shorter nonlinear load-deformation paths, due to an increase in $S_{a,n,vorh.}^{(i)}$ with a simultaneous decrease in $S_{a,n,max}^{(i)}$, leads to more and more minimal „dents“ or very fine kinks in the very short pushover curve segments. As a result, more and more adaptation steps are carried out in relation to the spectral step size $\Delta S_{a,B}^{(i)}$ – while the criterion values $k_{loc/glob}$ remain the same during the calculation. This is also not necessary „in terms of content“ and leads to an enormous, pointless additional computational effort.

  Initially, a scaling of the NL steps via the ratio of the current base shear increment $\Delta F_{b,max}^{(i)}$ to the increment $\Delta F_{b,max}^{(1)}$ of the very first „Max computation“ was planned, but this proved not to be expedient due to the strong slope differences (and the associated various segments with large $\Delta u_{2,mp}^{(i)}$ despite $\Delta Fb_{,max}^{(i)}$ of approximately 0). The NL steps are thus now scaled using the following relationship:

$$n_{steps,max\,comp.}^{(i)} = n_{steps,max\,comp.}^{(i-1)} \cdot \left(F_b^{(i-1)} < F_n < F_{b,max}^{(i)}\right) + 1 \qquad (3)$$

  In other words, the required number of calculation steps for the next „Max computation“ is estimated based on the part of the path of the previous „Max computation“ that would not have exceeded the new $F_{B,max}^{(i)}$ value and is larger than the last given value. Furthermore, one step is added for this limit value. This approach is based on the assumption that the „Max computation“ of step *i-1* will differ only slightly from the qualitative or quantitative progression up to the point at $F_{B,max}^{(i)}$ from the progression of the pending „Max computation“ of the current step *i*, so that it is at least senseful for estimating a suitable number of steps. For the specification of the „Min“ & „Max Num States“ in SAP2000, the following is now defined: $MinNumState = n_{steps,max\,comp.}^{(i)}$, $MaxNumState = 1.1 \cdot n_{steps,max\,comp.}^{(i)}$.
  In addition, a fixed value of 20 pushover steps is provided for the first correction calculation, which generally involves a very large increase in load and deformation; all further correction calculations are carried out with 5 pushover steps.
- In an adaptive „standard“ pushover calculation, all steps (of the max and correction calculations) are carried out with the step numbers defined in *arg.comp.nl_steps*, as the calculation time does not play such a significant role here.

Dealing with non reached $F_{b,max}^{(1)}$ in the „standard“ and „ami_o“ procedure:
In some cases of an „ami_o“ calculation, it may occur that the maximum base shear is not reached in the first „Max computation“ (sometimes by far) because the system cannot absorb this high load. In most cases, this is not a problem, as the actual base shear required in the performance state is much lower than the value determined on the basis of the elastic system properties in the first AMI step. However, there are two reasons why it makes sense or is algorithmically necessary to take into account the value $F_{b,(end)}^{(1)}$ actually achieved in the first step (with $F_{b,(end)}^{(1)} < F_{b,max,target}^{(1)}$) in the course of the further calculation steps:

1. In this case, it makes sense not to aim for $F_{b,max}^{(i)}$ "blindly" in the further calculation steps, but to use the minimum of $F_{b,max}^{(i)}$ and $F_{b,(end)}^{(1)}$, so that SAP2000 does not put unnecessary effort and thus computing time into the attempt to achieve a base shear that is now known to be unattainable.
2. Furthermore, the determination of suitable numbers of pushover steps described in the previous paragraph would then relate this number to an increment that is not calculated at all at the end. Here, it makes again more sense to refer to the actual upcoming path.

For these reasons, the new (current) load matrix *f_matrix* is scaled down at the end of the *caap_pushover_pointloads* function so that the sum of the load increments in the primary earthquake direction (current base shear increment $\Delta F_b^{(i)}$) together with the last existing base shear $F_{b,(end)}^{(i-1)}$ does not exceed the value $F_{b,(end)}^{(1)}$ and the latter is thus aimed for in case of doubt.

Dealing with negative natural frequencies of considered modes:
During the formation of plastic zones, it can sometimes happen that parts of stiffness-relevant areas or members become theoretically kinematic (kinematic substructures), which can lead to negative natural frequencies overall, although these are not critical in every case.

Within the CAAP tool, this is handled as follows:

1. In the case of a mode change assignment to a current mode with a negative natural frequency, the algorithm requires a corresponding confirmation by the user.
2. In the general consideration of supposedly (partially) kinematic modes (i.e. also in the development of a negative natural frequency of a non-changed mode), the user has two options:
   - Assignment to another mode (with positive natural frequency)
   - Continuation of the calculation without the same reassignment; in this case, the negative sign of the natural frequency or period is removed in order to avoid complex numbers for intermodal correlation factors and thus also complex numbers within the load vector, which would lead to a calculation crash in SAP2000. *It has been shown that in some cases, negative natural frequencies suddenly arise which, on the one hand, are very close in magnitude to the previous positive natural frequency and, on the other hand, correspond to natural modes that only exhibit minimal and therefore globally uncritical kinematic subvibrations. In such cases, a further calculation with this mode and the amount of the (actually) negative natural frequency is permissible and also appropriate.*