In the following function, we construct the residual needed for time integration of

second-order system

$$\mathbf{M}_\mathbf{V}\ddot{\mathbf{q}} + \mathbf{C}_\mathbf{V}\dot{\mathbf{q}} + \mathbf{F}_\mathbf{V}(\mathbf{q}) = \mathbf{V}^\top \mathbf{F}_{ext}(t),$$

where $\mathbf{M}_\mathbf{V} := \mathbf{V}^\top\mathbf{M}\mathbf{V},\ \mathbf{C}_\mathbf{V} := \mathbf{V}^\top\mathbf{C}\mathbf{V},\ \mathbf{F}_\mathbf{V}(\mathbf{q}) := \mathbf{V}^\top\mathbf{F}(\mathbf{V}\mathbf{q})$ are reduced operators. We use the reduced residual is defined as

$$\mathbf{r}(\ddot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{q}) = \mathbf{M}_\mathbf{V}\ddot{\mathbf{q}} + \mathbf{C}_\mathbf{V}\dot{\mathbf{q}} + \mathbf{F}_\mathbf{V}(\mathbf{q}) - \mathbf{V}^\top \mathbf{F}_{ext}(t).$$

A generic Residual function, whose *handle* is passed for performing Implicit Newmark and Generalized- $\alpha$ nonlinear time integration schemes in this code has the following syntax:

```
[r, drdqdd, drdqd, drdq, c0] = Residual(q,qd,qdd,t);
```

where

$$\begin{aligned}
\texttt{r} &= \mathbf{r},\\
\texttt{drdqdd} &= \frac{\partial \mathbf{r}}{\partial \ddot{\mathbf{q}}},\\
\texttt{drdqd} &= \frac{\partial \mathbf{r}}{\partial \dot{\mathbf{q}}},\\
\texttt{drdq} &= \frac{\partial \mathbf{r}}{\partial \mathbf{q}},\\
\texttt{c0} &= \text{a scalar measure for comparing the residual norm while checking for convergence}
\end{aligned}$$

The extra arguments:

1. `Assembly`, which is an instance of ReducedAssembly class
2. `Fext`, which is a function handle for the external forcing,
3. `V` is the basis used for Galerkin projection $\mathbf{V}$ and is a property of the ReducedAssembly Class

are required for computing the residual in this case.

New residual functions that follow the above-mentioned syntax can be written according to user preference. This way, the same time integration class can be used to solve a variety of problems.

Please refer to the Mechanical directory in the examples folder to understand applications and usage.

```
function [ r, drdqdd,drdqd,drdq, c0] = residual_reduced_nonlinear( q, qd, qdd, t, Assembly, Fex
```

In this function, it is assumed that the matrices $\mathbf{M}_\mathbf{V}, \mathbf{C}_\mathbf{V}$ for the finite element mesh were precomputed and stored in the `DATA` property of the `Assembly` object to avoid unnecessary assembly during each time-step.

```
V = Assembly.V;
M_V = Assembly.DATA.M;
C_V = Assembly.DATA.C;
```

The reduced tangent stiffness $\mathbf{K}_\mathbf{V} = \dfrac{\partial \mathbf{F}_\mathbf{V}}{\partial \mathbf{q}}$ and the the reduced internal force $\mathbf{F}_\mathbf{V}$, however, need to be assmbled at each iteration depending on the current state. This assembly is best perfomed at an element level in its reduced form as

$$\begin{aligned}\mathbf{K}_\mathbf{V}(\mathbf{q}) &= \textstyle\sum_{e=1}^{n_e} \mathbf{V}_e^\top \mathbf{K}_e(\mathbf{V}_e\mathbf{q})\mathbf{V}_e,\\ \mathbf{F}_\mathbf{V}(\mathbf{q}) &= \textstyle\sum_{e=1}^{n_e} \mathbf{V}_e^\top \mathbf{F}_e(\mathbf{V}_e\mathbf{q})\end{aligned}.$$

We first obtain the full degrees of freedom vector $\mathbf{u}$ over the mesh from the reduced variables $\mathbf{q}$ as $\mathbf{u} = \mathbf{V}\mathbf{q}$, and then directly assemble the reduced nonlinear operators with the Assembly class method `tangent_stiffness_and_force_modal(u,V)`.

```
u = V*q;
[K_V, F_V] = Assembly.tangent_stiffness_and_force(u);
```

Residual is computed according to the formula above:

```
F_inertial = M_V * qdd;
F_damping = C_V * qd;
F_ext_V =  V.'*Fext(t);

r = F_inertial + F_damping + F_V - F_ext_V ;

drdqdd = M_V;
drdqd = C_V;
drdq = K_V;
```

We use the following measure to comapre the norm of the residual $\mathbf{r}$

`c0` $= \|\mathbf{M}_\mathbf{V}\ddot{\mathbf{q}}\| + \|\mathbf{C}_\mathbf{V}\dot{\mathbf{q}}\| + \|\mathbf{F}_\mathbf{V}(\mathbf{q})\| + \|\mathbf{V}^\top\mathbf{F}_{ext}(t)\|$

```
c0 = norm(F_inertial) + norm(F_damping) + norm(F_V) + norm(F_ext_V);

end
```