fix adapt command

Syntax:

```
fix ID group-ID adapt N attribute args ... keyword value ...
```

- ID, group-ID are documented in fix command
- adapt = style name of this fix command
- N = adapt simulation settings every this many timesteps
- one or more attribute/arg pairs may be appended
- attribute = *pair* or *bond* or *kspace* or *atom*

```
  pair args = pstyle pparam I J v_name
      pstyle = pair style name, e.g. lj/cut
      pparam = parameter to adapt over time
      I,J = type pair(s) to set parameter for
      v_name = variable with name that calculates value of pparam
    bond args = bstyle bparam I v_name
      bstyle = bond style name, e.g. harmonic
      bparam = parameter to adapt over time
      I = type bond to set parameter for
      v_name = variable with name that calculates value of bparam
    kspace arg = v_name
      v_name = variable with name that calculates scale factor on K-space terms
    atom args = aparam v_name
      aparam = parameter to adapt over time
      v_name = variable with name that calculates value of aparam
```

- zero or more keyword/value pairs may be appended
- keyword = *scale* or *reset* or *fscale*

```
  scale value = no or yes
      no = the variable value is the new setting
      yes = the variable value multiplies the original setting
    reset value = no or yes
      no = values will remain altered at the end of a run
      yes = reset altered values to their original values at the end of a run
    fscale value = no or yes
      no = energy and forces will be adapted during the run
      yes = only forces will be adapted during the run
```

Examples:

```
fix 1 all adapt 1 pair soft a 1 1 v_prefactor
fix 1 all adapt 1 pair soft a 2* 3 v_prefactor
fix 1 all adapt 1 pair lj/cut epsilon * * v_scale1 coul/cut scale 3 3 v_scale2 scale yes reset
fix 1 all adapt 10 atom diameter v_size

fix 1 all adapt 1 pair lj/cut/tip4p/long fscale * * v_prefactor kspace v_prefactor fscale yes

variable ramp_up equal "ramp(0.01,0.5)"
fix stretch all adapt 1 bond harmonic r0 1 v_ramp_up
```

Description:

Change or adapt one or more specific simulation attributes or settings over time as a simulation runs. Pair potential and K-space and atom attributes which can be varied by this fix are discussed below. Many other fixes can also be used to time-vary simulation parameters, e.g. the "fix deform" command will change the simulation box size/shape and the "fix move" command will change atom positions and velocities in a prescribed manner. Also note that many commands allow variables as arguments for specific parameters, if described in that manner on their doc pages. An equal-style variable can calculate a time-dependent quantity, so this is another way to vary a simulation parameter over time.

If *N* is specified as 0, the specified attributes are only changed once, before the simulation begins. This is all that is needed if the associated variables are not time-dependent. If *N* > 0, then changes are made every *N* steps during the simulation, presumably with a variable that is time-dependent.

Depending on the value of the *reset* keyword, attributes changed by this fix will or will not be reset back to their original values at the end of a simulation. Even if *reset* is specified as *yes*, a restart file written during a simulation will contain the modified settings.

If the *scale* keyword is set to *no*, then the value the parameter is set to will be whatever the variable generates. If the *scale* keyword is set to *yes*, then the value of the altered parameter will be the initial value of that parameter multiplied by whatever the variable generates. I.e. the variable is now a "scale factor" applied in (presumably) a time-varying fashion to the parameter.

Note that whether scale is *no* or *yes*, internally, the parameters themselves are actually altered by this fix. Make sure you use the *reset yes* option if you want the parameters to be restored to their initial values after the run.

The *pair* keyword enables various parameters of potentials defined by the pair_style command to be changed, if the pair style supports it. Note that the pair_style and pair_coeff commands must be used in the usual manner to specify these parameters initially; the fix adapt command simply overrides the parameters.

The *pstyle* argument is the name of the pair style. If pair_style hybrid or hybrid/overlay is used, *pstyle* should be a sub-style name. If there are multiple sub-styles using the same pair style, then *pstyle* should be specified as "style:N" where N is which instance of the pair style you wish to adapt, e.g. the first, second, etc. For example, *pstyle* could be specified as "soft" or "lubricate" or "lj/cut:1" or "lj/cut:2". The *pparam* argument is the name of the parameter to change. This is the current list of pair styles and parameters that can be varied by this fix. See the doc pages for individual pair styles and their energy formulas for the meaning of these parameters:

born	a,b,c	type pairs
buck	a,c	type pairs
coul/cut	scale	type pairs
coul/debye	scale	type pairs
coul/long	scale	type pairs

eam, eam/alloy, eam/fs	scale	type pairs
gauss	a	type pairs
kim	PARAM_FREE_*&#58i,j,...	global
lj/cut	epsilon,sigma	type pairs
lj/cut/coul/long	epsilon,sigma,fscale	type pairs
lj/cut/tip4p/long	epsilon,sigma,fscale	type pairs
lj/expand	epsilon,sigma,delta	type pairs
lj/sf/dipole/sf	epsilon,sigma,scale	type pairs
lubricate	mu	global
meam/c	fscale	type pairs
morse	d0,r0,alpha	type pairs
soft	a	type pairs
sw	fscale	type pairs
ufm,ufm/rw	epsilon,sigma,fscale	type pairs

NOTE: It is easy to add new pairwise potentials and their parameters to this list. All it typically takes is adding an extract() method to the pair_*.cpp file associated with the potential.

Some parameters are global settings for the pair style, e.g. the viscosity setting "mu" for pair_style lubricate. For pair_kim, all free parameters supported by the KIM Model are available (e.g., PARAM_FREE_sigmas provided by the LennardJones612_Universal__MO_826355984548_001 Model). If the free parameter corresponds to an array, then the particular array element to be adapted must be specified (e.g., "PARAM_FREE_sigmas:10", to adapt the tenth entry of the sigmas array). Other parameters apply to atom type pairs within the pair style, e.g. the prefactor "a" for pair_style soft.

Note that for many of the potentials, the parameter that can be varied is effectively a prefactor on the entire energy expression for the potential, e.g. the lj/cut epsilon. The parameters listed as "scale" are exactly that, since the energy expression for the coul/cut potential (for example) has no labeled prefactor in its formula. To apply an effective prefactor to some potentials, multiple parameters need to be altered. For example, the Buckingham potential needs both the A and C terms altered together. To scale the Buckingham potential, you should thus list the pair style twice, once for A and once for C.

If a type pair parameter is specified, the *I* and *J* settings should be specified to indicate which type pairs to apply it to. If a global parameter is specified, the *I* and *J* settings still need to be specified, but are ignored.

Similar to the pair_coeff command, I and J can be specified in one of two ways. Explicit numeric values can be used for each, as in the 1st example above. I <= J is required. LAMMPS sets the coefficients for the symmetric J,I interaction to the same values.

A wild-card asterisk can be used in place of or in conjunction with the I,J arguments to set the coefficients for multiple pairs of atom types. This takes the form "*" or "*n" or "n*" or "m*n". If N = the number of atom types, then an asterisk with no numeric

values means all types from 1 to N. A leading asterisk means all types from 1 to n (inclusive). A trailing asterisk means all types from n to N (inclusive). A middle asterisk means all types from m to n (inclusive). Note that only type pairs with I <= J are considered; if asterisks imply type pairs where J < I, they are ignored.

IMPORTANT NOTE: If pair_style hybrid or hybrid/overlay is being used, then the *pstyle* will be a sub-style name. You must specify I,J arguments that correspond to type pair values defined (via the pair_coeff command) for that sub-style.

The *v_name* argument for keyword *pair* is the name of an equal-style variable which will be evaluated each time this fix is invoked to set the parameter to a new value. It should be specified as v_name, where name is the variable name. Equal-style variables can specify formulas with various mathematical functions, and include thermo_style command keywords for the simulation box parameters and timestep and elapsed time. Thus it is easy to specify parameters that change as a function of time or span consecutive runs in a continuous fashion. For the latter, see the *start* and *stop* keywords of the run command and the *elaplong* keyword of thermo_style custom for details.

For example, these commands would change the prefactor coefficient of the pair_style soft potential from 10.0 to 30.0 in a linear fashion over the course of a simulation:

```
variable prefactor equal ramp(10,30)
fix 1 all adapt 1 pair soft a * * v_prefactor
```

The *bond* keyword uses the specified variable to change the value of a bond coefficient over time, very similar to how the *pair* keyword operates. The only difference is that now a bond coefficient for a given bond type is adapted.

A wild-card asterisk can be used in place of or in conjunction with the bond type argument to set the coefficients for multiple bond types. This takes the form "*" or "*n" or "n*" or "m*n". If N = the number of atom types, then an asterisk with no numeric values means all types from 1 to N. A leading asterisk means all types from 1 to n (inclusive). A trailing asterisk means all types from n to N (inclusive). A middle asterisk means all types from m to n (inclusive).

Currently *bond* does not support bond_style hybrid nor bond_style hybrid/overlay as bond styles. The only bonds that currently are working with fix_adapt are

harmonic	k,r0	type bonds

The *kspace* keyword used the specified variable as a scale factor on the energy, forces, virial calculated by whatever K-Space solver is defined by the kspace_style command. If the variable has a value of 1.0, then the solver is unaltered.

NOTE: If the *fscale* keyword is set to no or is not specified for *pppm* or *pppm/tip4p*, then energy and forces will be scaled by the specified variable as a scale factor (default). If the *fscale* keyword is set to yes, then only the forces will be scaled by the specified variable.

The *kspace* keyword works this way whether the *scale* keyword is set to *no* or *yes*.

The *atom* keyword enables various atom properties to be changed. The *aparam* argument is the name of the parameter to change. This is the current list of atom parameters that can be varied by this fix:

- charge = charge on particle
- diameter = diameter of particle

The *v_name* argument of the *atom* keyword is the name of an equal-style variable which will be evaluated each time this fix is invoked to set the parameter to a new value. It should be specified as v_name, where name is the variable name. See the discussion above describing the formulas associated with equal-style variables. The new value is assigned to the corresponding attribute for all atoms in the fix group.

NOTE: The *atom* keyword works this way whether the *scale* keyword is set to *no* or *yes*. I.e. the use of scale yes is not yet supported by the *atom* keyword.

If the atom parameter is *diameter* and per-atom density and per-atom mass are defined for particles (e.g. atom_style granular), then the mass of each particle is also changed when the diameter changes (density is assumed to stay constant).

For example, these commands would shrink the diameter of all granular particles in the "center" group from 1.0 to 0.1 in a linear fashion over the course of a 1000-step simulation:

```
variable size equal ramp(1.0,0.1)
fix 1 center adapt 10 atom diameter v_size
```

Restart, fix_modify, output, run start/stop, minimize info:

No information about this fix is written to binary restart files. None of the fix_modify options are relevant to this fix. No global or per-atom quantities are stored by this fix for access by various output commands. No parameter of this fix can be used with the *start/stop* keywords of the run command. This fix is not invoked during energy minimization.

For rRESPA time integration, this fix changes parameters on the outermost rRESPA level.

Restrictions: *fscale* keyword can only be used with *pppm* and *pppm/tip4p* styles.

Related commands:

compute ti

Default:

The option defaults are scale = no, reset = no, fscale = no.