

sweepRs (generic function with 1 method)

```
1 #####sweep particle radius and output force matrix to file#####
2 function sweepRs(parPosTmp::Matrix{Float64},Rs::Float64)
3 parPos=deepcopy(parPosTmp);
4 dRs=0.0001;
5 println("Input Data file name and press Enter:");
6 fileName=readline();
7 outputData=open(fileName*".txt","w");
8 totRndNo=30;
9 rndDigNo=9;
10 for iTmp in 1:totRndNo
11 write(outputData,"round"*string(iTmp));
12 println("round"*string(iTmp));
13 write(outputData,"\n");
14 write(outputData,"Rs: "*string(Rs));
15 write(outputData,"\n");
16 write(outputData,"initial positions: ");
17 write(outputData,"\n");
18 writedlm(outputData,parPos,'\t');
19 write(outputData,"\n");
20 parPos=md(parPos,Rs);
21 write(outputData,"equilibrium positions: ");
22 write(outputData,"\n");
23 writedlm(outputData,parPos,'\t');
24 write(outputData,"\n");
25 kMat=forceMat(parPos,Rs);
26 write(outputData,"force matrix: ");
27 write(outputData,"\n");
28 writedlm(outputData,kMat,'\t');
29 println(kMat);
30 write(outputData,"\n");
31 egv=eigen(kMat);
32 write(outputData,"eigenvalues: ");
33 write(outputData,"\n");
34 writedlm(outputData,round.(egv.values;digits=rndDigNo),'\t');
35 println(round.(egv.values;digits=rndDigNo));
36 write(outputData,"\n");
37 write(outputData,"eigenvectors: ");
38 write(outputData,"\n");
39 writedlm(outputData,round.(egv.vectors;digits=rndDigNo),'\t');
40 write(outputData,"\n");
41 write(outputData,"\n");
42 Rs=Rs+dRs;
43 flush(outputData);
44 end
45 close(outputData);
46 return 0;
47 end
```

sweepRsS (generic function with 1 method)

```
1 #####sweep particle radius and output force matrix to file#####
2 function
3 sweepRsS(center::Bool,pNo::Integer,foldNo::Integer,ringNo::Integer,Rc::Array{Float64}
4 ,Rs::Float64)
5 dRs=0.0001;
6 println("Input Data file name and press Enter:");
7 fileName=readline();
8 outputData=open(fileName*".txt","w");
9 totRndNo=30;
10 rndDigNo=12;
11 rcTmp=Rc;
12 parPos=parPosGen(center,pNo,foldNo,ringNo,rcTmp);
13 for iTmp in 1:totRndNo
14 write(outputData,"round"*string(iTmp));
15 println("round"*string(iTmp));
16 write(outputData,"\n");
17 write(outputData,"Rs: "*string(Rs));
18 write(outputData,"\n");
19 write(outputData,"initial positions: ");
20 write(outputData,"\n");
21 writedlm(outputData,parPos,'\t');
22 write(outputData,"\n");
23 rcTmp=mdS(center,pNo,foldNo,ringNo,rcTmp,Rs);
24 parPos=parPosGen(center,pNo,foldNo,ringNo,rcTmp);
25 write(outputData,"equilibrium positions: ");
26 write(outputData,"\n");
27 writedlm(outputData,parPos,'\t');
28 write(outputData,"\n");
29 kMat=forceMat(parPos,Rs);
30 write(outputData,"force matrix: ");
31 write(outputData,"\n");
32 writedlm(outputData,kMat,'\t');
33 println(kMat);
34 write(outputData,"\n");
35 egv=eigen(kMat);
36 write(outputData,"eigenvalues: ");
37 write(outputData,"\n");
38 writedlm(outputData,round.(egv.values;digits=rndDigNo),'\t');
39 println(round.(egv.values;digits=rndDigNo));
40 write(outputData,"\n");
41 write(outputData,"eigenvectors: ");
42 write(outputData,"\n");
43 writedlm(outputData,round.(egv.vectors;digits=rndDigNo),'\t');
44 write(outputData,"\n");
45 write(outputData,"\n");
46 Rs=Rs+dRs;
47 flush(outputData);
48 end
49 close(outputData);
50 return 0;
51 end
52
```

