Electronic Supplementary Information for the paper: "Computational Design of the Inorganic Nonlinear Optical Crystals Based on the Genetic Algorithm"

Zhenxing Fang,[a] Jing Lin,[a] Rong Liu,[b] Ping Liu,[a] Yi Li,*[a] Xin Huang,[a] Kaining Ding,[a] Lixin Ning,[c] and Yongfan Zhang*[a,d]

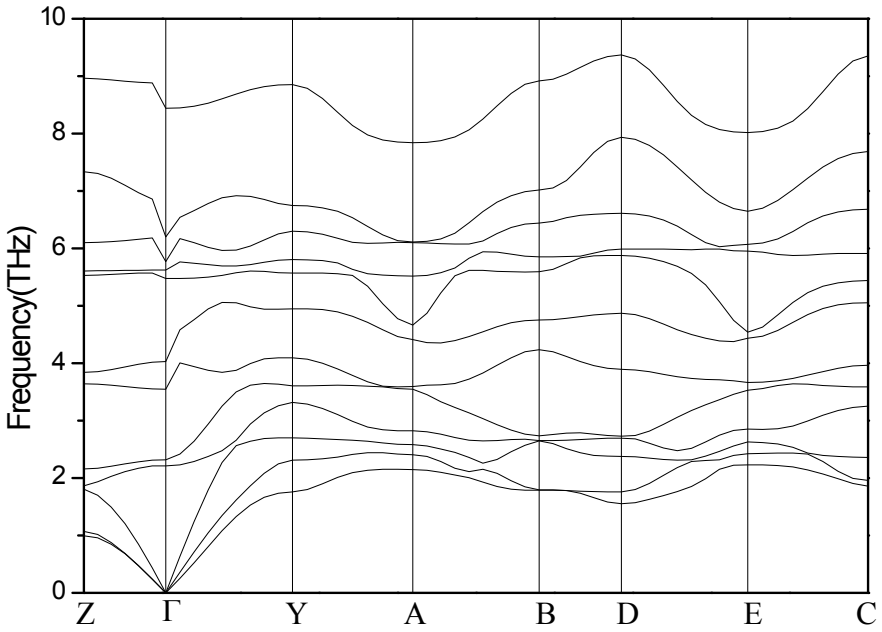
[a] *College of Chemistry, Fuzhou University, Fuzhou, Fujian, 350116, China*

[b] *College of Mathematics and Computer Science, Fuzhou University, Fuzhou, Fujian, 350116, China*

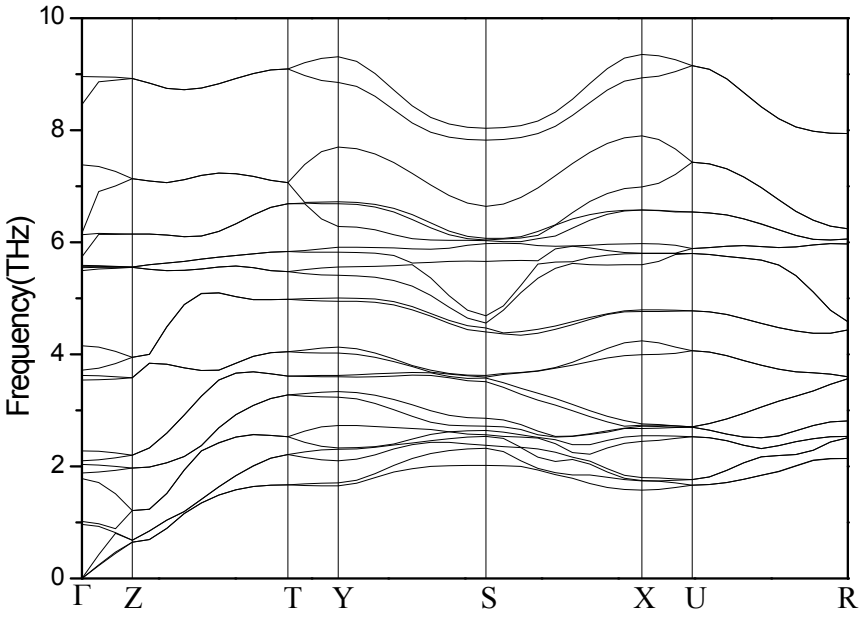
[c] *Department of Physics, Anhui Normal University, Wuhu, Anhui, 241000, China*

[d] *Key Laboratory of Optoelectronic Materials Chemistry and Physics, Chinese Academy of Sciences, Fuzhou, Fujian, 350002, China*

Figure S1:



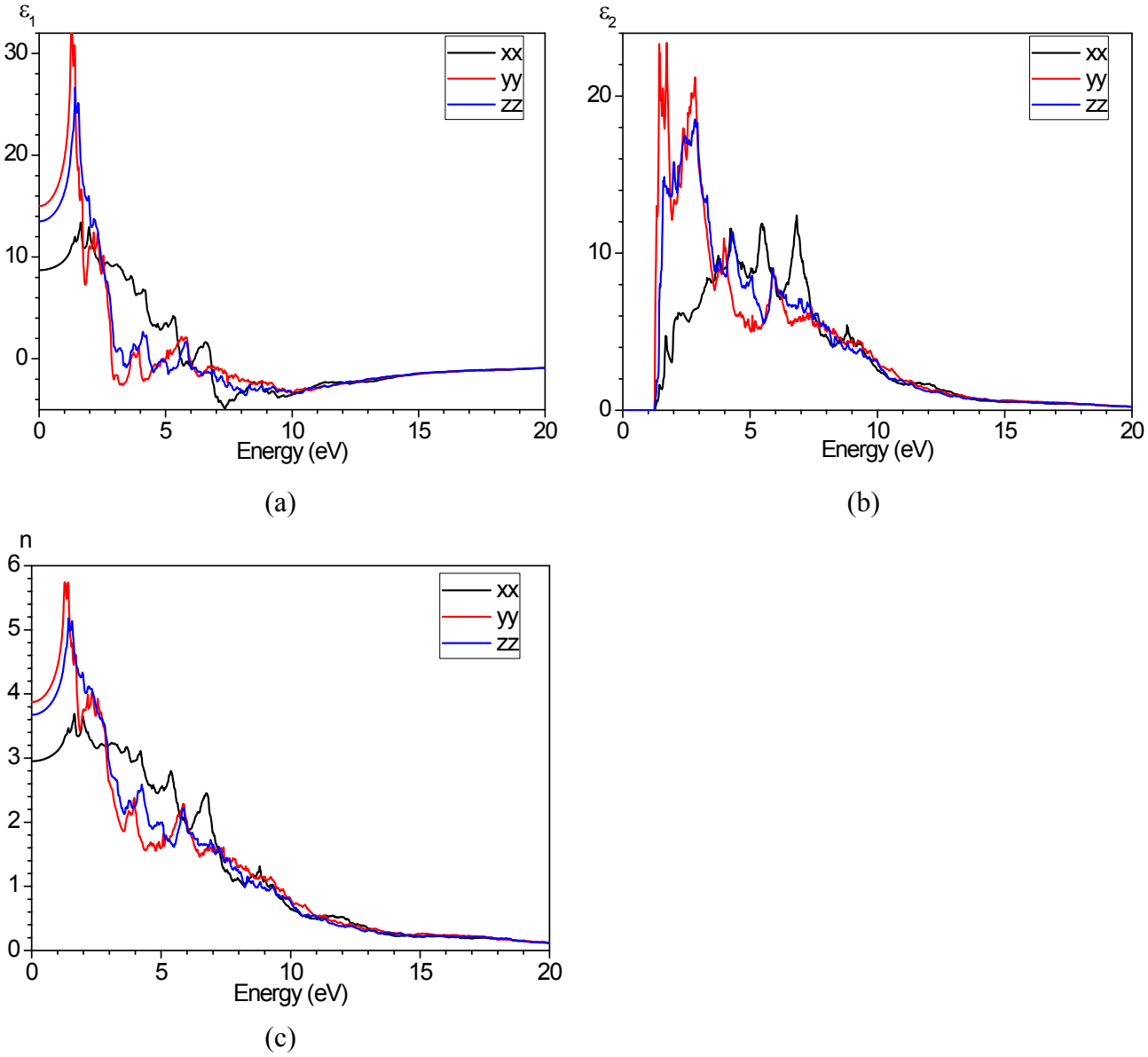
(a)



(b)

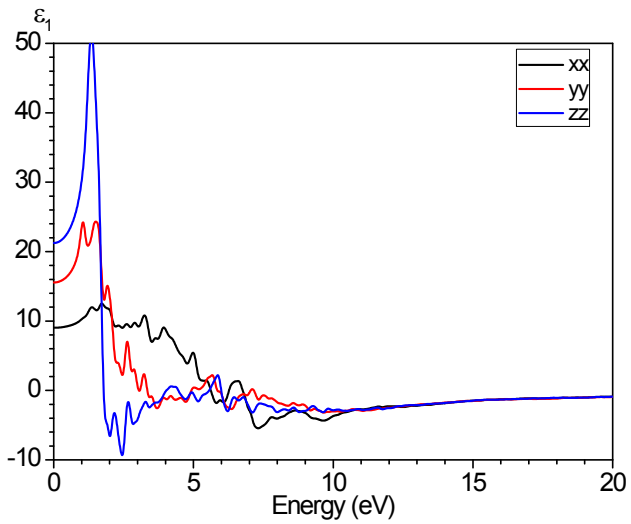
Calculated phonon dispersion curves of $LiAsSe_2$ crystallized in the (a) *Pm* space group, and (b) *Pmc2₁* space group

Figure S2:

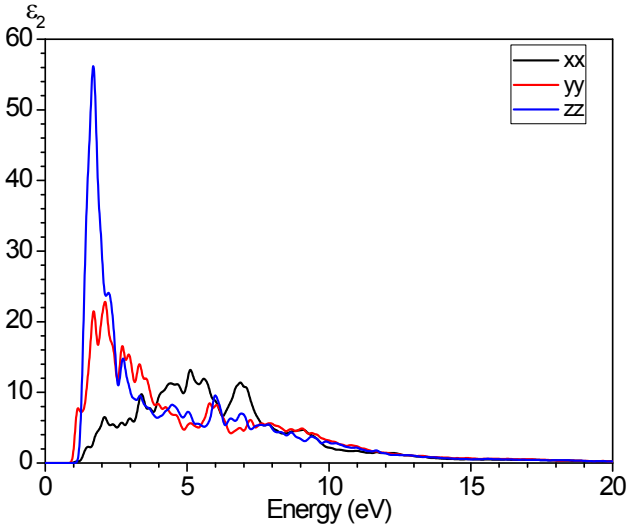


(a) Real part and (b) imaginary part of the calculated complex dielectric functions, and (c) the refractive indices of $LiAsSe_2$ crystallized in the *Cc* space group.

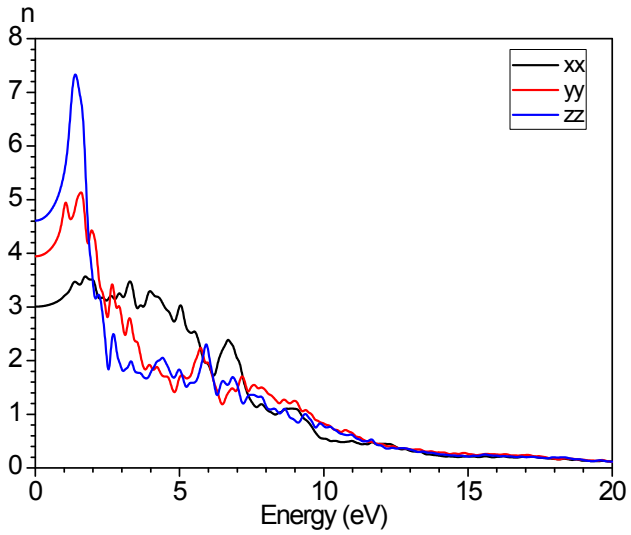
Figure S3:



(a)



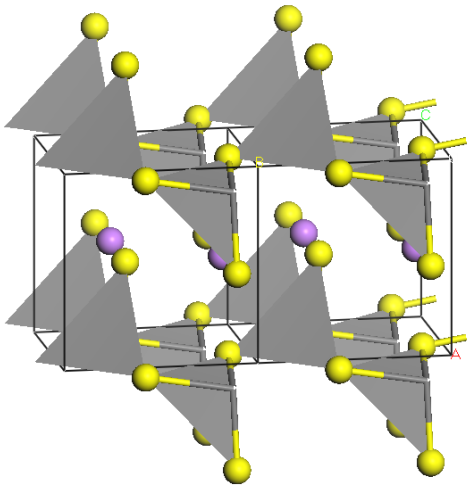
(b)



(c)

(a) Real part and (b) imaginary part of the calculated complex dielectric functions, and (c) the refractive indices of $LiAsSe_2$ crystallized in the *Pmc2₁* space group.

Table S1 Lattice constants and fractional atomic positions of the $LiAsSe_2$ crystallized in the $P2_1$ space group



Spacegroup number	4
a (Å)	5.793
b (Å)	5.559
c (Å)	5.632
β (degree)	86.29
Li	0.49974, 0.14719, 0.25393
As	0.08354, 0.16281, 0.79262
Se	0.00373, 0.61705, 0.76730
Se	0.49075, 0.14741, 0.74962
Volume per formula ($Å^3$)	90.50
As – terminal Se (Å)	2.357
As – bridging Se (Å)	2.573, 2.510

Part of the source code used in the present work:

(1) tetrahedron.m

```
%This function is used to create a tetrahedral coordination environment of each cation
function [positions]= tetrahedron(cell,cation_positions,anion_ positions)

%An example template for a tetrahedron, cation is at (0,0,0), and one anion atom is set at the z-axis.
%In here, the cation-anion distance is set to 2.26 Å(the sum of the covalent radii of Ga and S atoms)
origin_x=[0.0,0.0,2.26 ;0.0,-2.1307,-0.7533 ;-1.8453,1.0654,-0.7533 ;1.8453,1.0654, -0.7533];
%Number of cations
[cation_number,col]=size(cation_positions);
%Number of anions of each cation
[anion_number,col]=size(origin_x);
for k=1:cation_number
% Obtain the Cartesian coordinates of each cation atom
    x_cation=cation_positions(k,:)*cell;
% Create the rotation angles randomly
    sita=pi*rand(1,1);
    fai=2*pi*rand(1,1);
% Rotate around the z-axis first
    for i=1:anion_number
        x1(i,1)=origin_x(i,1)*cos(fai)-origin_x(i,2)*sin(fai);
        x1(i,2)=origin_x(i,1)*sin(fai)+origin_x(i,2)*cos(fai);
        x1(i,3)=origin_x(i,3);
    end
% Then rotating around the x-axis
    for i=1:anion_number
        x2(i,1)=x1(i,1);
        x2(i,2)=x1(i,2)*cos(sita)-x1(i,3)*sin(sita);
        x2(i,3)=x1(i,2)*sin(sita)+x1(i,3)*cos(sita);
    end
% Now the Cartesian coordinates of new tetrahedron are created
% Transfer the coordinates to fractional
    for i=1:anion_number
        xyz(i,:)=x2(i,:)+x_cation;
        xnew(i,:)=xyz(i,:)*inv(cell);
        xnew(i,1)=rem(xnew(i,1),1);
        xnew(i,2)=rem(xnew(i,2),1);
        xnew(i,3)=rem(xnew(i,3),1);
%       Take care the negative fractional coordinates
        for j=1:3
            if xnew(i,j)< 0
                xnew(i,j)=xnew(i,j)+1.0;
            end
```

```
        end
    end
% Final coordinates of anions
    anion_positions((k-1)*anion_number+1:k*anion_number,:)=xnew(1:anion_number,:);
end
```

(2) cal_energy_list.m

%This function is used to construct POSCAR, KPOINTS files, and obtain the structure and energy calculated by VASP

```
function [energy_list,optlat,optcoor]=cal_energy_list(range,crystal_stru_array,step,numatom,
atomtype, kspace,nsteps,source,command,tol)
if ndims(crystal_stru_array)==2
    p=1;
    [row,col]=size(crystal_stru_array);
else
    [row,col,p]=size(crystal_stru_array);
end
energy_list=zeros(1,p, 'double');
for i=1:p
% Create the POSCAR file
  fid = fopen('calc/POSCAR', 'w+');
  fprintf(fid,'%s %d\n','EA',(step-1)*p+i);
  fprintf(fid,'%s\n','1.0');
  for j=1:3
      for k=1:3
          fprintf(fid,' %1.16f', range(j, k, i));
      end
      fprintf(fid,'\n');
  end
  volume_begin=det(range(:,:,i));
  for j=1:atomtype
      fprintf(fid,' %d',numatom(j));
  end
  fprintf(fid,'\n');
  fprintf(fid,'%s\n','Dir');
  if p==1
    for j = 1:row
        for k = 1:col
            fprintf(fid,' %1.16f', crystal_stru_array(j, k));
        end
        fprintf(fid,'\n');
    end
  else
```

```
        for j = 1:row
            for k = 1:col
                fprintf(fid,' %1.16f', crystal_stru_array(j, k,i));
            end
            fprintf(fid,'\n');
        end
    end
    fclose(fid);
    unix(['cat calc/POSCAR >> save_POSCAR']);
    time=0;
    tic;
%Note: for the most stable structure of previous generation, it only needs to be
%recalculated again by using the setting of last step.
    if source(i,:)==' keep old'
        nstart=nsteps;
    else
         nstart=1;
    end
    for j=nstart:nsteps
        unix(['cat INCAR_' num2str(j) ' > calc/INCAR']);
%    Create the KPOINTS file
        if kspace(j)>0
%         Automatically generate KPOINTS file
            lattice = lat_convert(range(:,:,i));
            vol = abs(det(range(:,:,i)));
            dist = zeros(1,3);
            dist(3) = vol/(lattice(1)*lattice(2)*sin(lattice(6)));
            dist(2) = vol/(lattice(1)*lattice(3)*sin(lattice(5)));
            dist(1) = vol/(lattice(2)*lattice(3)*sin(lattice(4)));
            Kpoints = ceil(1./(dist*kspace(j)));
%         Note: generally, the level of the last step is high, therefore, the corresponding
%         K point required is larger than 1. If it equals to 1, we set it to 2.
            if j==nsteps
                for ik=1:3
                    if Kpoints(ik)==1
                        Kpoints(ik)=2;
                    end
                end
            end
            unix('cat /dev/null > calc/KPOINTS');
            unix(['echo ' 'EA' ' >>calc/KPOINTS']);
            unix(['echo ' '0' ' >>calc/KPOINTS']);
            unix(['echo ' 'Gamma' ' >>calc/KPOINTS']);
            unix(['echo ' num2str(Kpoints(1,:)) ' >>calc/KPOINTS']);
```

```
      else
%       Use the KPOINTS defined by user
        unix(['cat KPOINTS_0 > calc/KPOINTS']);
        Kpoints(1,:)=[0 0 0];
      end
      unix('cp POTCAR calc');
      unix(command{j});
      [nothing,result]=unix('./getEnergy');
      e(j)=str2double(result);
      [nothing,result]=unix('./getStep');
      ns(j)=str2num(result);
      unix('cp calc/CONTCAR calc/POSCAR');
      end
      time=toc;
%   Read the new coordinates and lattice vectors from CONTCAR file and
%   save the CONTCAR to save_CONTCAR file
      unix(['cat calc/vasp.out' ' >    calc/scratch/vasp.out_' num2str((step-1)*p+i)]);
      unix(['cat calc/KPOINTS' ' >    calc/scratch/KPOINTS_' num2str((step-1)*p+i)]);
      unix(['cat calc/OUTCAR' ' >    calc/scratch/OUTCAR_' num2str((step-1)*p+i)]);
      unix(['cat calc/OSZICAR' ' >    calc/scratch/OSZICAR_' num2str((step-1)*p+i)]);
      unix(['cat calc/CONTCAR' ' >    calc/scratch/CONTCAR_' num2str((step-1)*p+i)]);
%   Determine the space group of current structure
      unix(['echo ' num2str(atomtype,10) '    > sg.in']);
      unix(['echo ' num2str(tol,10) ' >> sg.in']);
      unix('cp calc/CONTCAR .');
      [nothing,sg]=unix('vasp_space_group>sg.dat');
      [nothing,sg]=unix('grep Space sg.dat');
      unix('rm    CONTCAR sg.in sg.dat');
      [fid,message]=fopen('calc/CONTCAR', 'r');
      [fid1,message]=fopen('save_CONTCAR','a+');
      fprintf(fid1,'%s %d %s','EA',(step-1)*p+i,sg);
      tmp=fgetl(fid);
      fprintf(fid1,'%s\n',tmp);
      scale_factor=str2num(tmp);
      latt=zeros(3);
      for k=1:3
          tmp = fgetl(fid);
          latt(k,:)=str2num(tmp);
          fprintf(fid1,'%f %f %f\n',latt(k,:));
      end
      latt=latt*scale_factor;
      volume=det(latt);
      tmp=fgetl(fid);
      fprintf(fid1,'%s\n',tmp);
```

```
        ntype=str2num(tmp);
        if isempty(ntype)
           tmp = fgetl(fid);
           fprintf(fid1,'%s\n',tmp);
           ntype=str2num(tmp);
        end
        natom=sum(ntype);
        tmp_mode=fgetl(fid);
        fprintf(fid1,'%s\n',tmp_mode);
        if (tmp_mode(1) == 's') | (tmp_mode(1) == 'S')
           tmp_mode = fgetl(fid);
           fprintf(fid1,'%s\n',tmp_mode);
           sss=fscanf(fid,'%g %g %g %s %s %s',[6,natom]);
           fprintf(fid1,'%f %f %f %s %s %s\n',sss);
        else
           sss=fscanf(fid,'%g %g %g',[3,natom]);
           fprintf(fid1,'%f %f %f\n',sss);
        end
        ss=sss';
        coords= ss(:,1:3);
        fclose(fid);
        fclose(fid1);
        coords=coords-floor(coords);
        optlat(:,:,i)=latt;
        optcoor(:,:,i)=coords;
%    Determine the minimum distance among the current structure
        minimum=cell_distant(optlat(:,:,i),optcoor(:,:,i));
        [fid,message] = fopen('save_ENERGY', 'a+');
        energy_list(i)=e(nsteps);
        fprintf(fid,'%3d%5d%10s%15.5f%10.3f %1s%3d%3d%3d%1s %7.3f %5.0f    %s', step,
        (step-1)*p+i,source(i,:),energy_list(i),volume,'[',Kpoints,']',minimum,time,sg(13:length(sg)));
        fclose(fid);
        [fid,message] = fopen('save_fitlist', 'a+');
        fprintf(fid,'%s %d','EA',(step-1)*p+i);
        for j=nstart:nsteps
            fprintf(fid,'%s %d %s %d %f',' step',j,':',ns(j),e(j));
        end
        fprintf(fid,'\n');
        fclose(fid);
end
```