**Flux and Brightness Calculations
for Various Synchrotron Radiation Sources**

Jill M. Weber and Steven L. Hulbert

November 1991

NATIONAL SYNCHROTRON LIGHT SOURCE

**BROOKHAVEN NATIONAL LABORATORY
ASSOCIATED UNIVERSITIES, INC.**

Under Contract No. DE-AC02-76CH00016 with the

UNITED STATES DEPARTMENT OF ENERGY

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency, contractor or subcontractor thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency, contractor or subcontractor thereof.

BNL--47034

DE92 007615

**Flux and Brightness Calculations
for Various Synchrotron Radiation Sources**

Jill M. Weber and Steven L. Hulbert

November 1991

Research Supported by the
OFFICE OF BASIC ENERGY SCIENCES
U.S. DEPARTMENT OF ENERGY
WASHINGTON, D. C.

NATIONAL SYNCHROTRON LIGHT SOURCE
BROOKHAVEN NATIONAL LABORATORY
Associated Universities, Inc.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

I. Introduction

Synchrotron radiation (SR) storage rings are powerful scientific and technological tools. The first generation of storage rings in the U.S., e.g. SURF (Washington, D.C.), Tantalus (Wisconsin), SSRL (Stanford), and CHESS (Cornell), revolutionized VUV, soft X-ray, and hard X-ray science. The second (present) generation of storage rings, e.g. the NSLS VUV and XRAY rings and Aladdin (Wisconsin), have sustained the revolution by providing higher stored currents and up to a factor of ten smaller electron beam sizes than the first generation sources. This has made possible a large number of experiments that could not be performed using first generation sources. In addition, the NSLS XRAY ring design optimizes the performance of wigglers (high field periodic magnetic insertion devices). The third generation storage rings, e.g. ALS (Berkeley) and APS (Argonne), are being designed to optimize the performance of undulators (low field periodic magnetic insertion devices). These extremely high brightness sources will further revolutionize x-ray science by providing diffraction-limited x-ray beams.

The output of undulators and wigglers is distinct from that of bending magnets in magnitude, spectral shape, and in spatial and angular size. Using published equations, we have developed computer programs to calculate the flux, central intensity, and brightness output from bending magnets and selected wigglers and undulators of the NSLS VUV and XRAY rings, the Advanced Light Source (ALS), and the Advanced Photon Source (APS).

Following is a summary of the equations used, the graphs and data produced, and the computer codes written. These codes, written in the C programming language, can be used to calculate the flux, central intensity, and brightness curves for bending magnets and insertion devices on any storage ring.

II. Bending Magnets

Bending magnets produce the simplest form of synchrotron radiation from storage rings. The radiation emission properties in the ideal case, where angular divergence and emittance effects are negligible, were derived originally by Schwinger¹.

The angular distribution of radiation emitted by electrons moving through a bending magnet with a circular trajectory in the horizontal plane is given by

$$\frac{d^2 F_{bm}(\omega)}{d\theta d\psi} = \frac{3\alpha}{4\pi^2} \gamma^2 \frac{\Delta\omega}{\omega} \frac{I}{e} y^2 (1 + X^2)^2 \times \left[K_{2/3}^2(\xi) + \frac{X^2}{1 + X^2} K_{1/3}^2(\xi) \right], \quad (1)$$

where

- F_{bm} = photon flux (number of photons per second),
- θ = observation angle in the horizontal plane,
- ψ = observation angle in the vertical plane,
- α = fine structure constant $\equiv 1/137$,
- γ = electron energy / $m_e c^2$ (m_e = electron mass, c = velocity of light),
- ω = angular frequency of photons ($\hbar\omega$ = photon energy = ϵ),
- I = beam current,
- e = electron charge = 1.602×10^{-19} coulomb,
- $y = \omega / \omega_c = \epsilon / \epsilon_c$ (ω_c = critical frequency = $3\gamma^3 c/2\rho$),
- ρ = radius of instantaneous curvature of the electron trajectory = E/ecB ;
in practical units, $\rho(m) = 3.33 E(\text{GeV})/B(T)$,
- E = electron beam energy,
- B = magnetic field strength,
- $\epsilon_c = \hbar\omega_c$ [$\epsilon_c(\text{keV}) = 0.665 E^2(\text{GeV})B(T)$],
- $X = \gamma \psi$, and
- $\xi = y(1 + X^2)^{3/2} / 2$.

The subscripted K's are modified Bessel functions of the second kind evaluated as an infinite sum by Kostroun² in routine bend_magnet.c (see Bending Magnet in the Appendix). In the horizontal plane ($\psi = 0$), Eq. (1) becomes

$$\left. \frac{d^2 F_{bm}(y)}{d\theta d\psi} \right|_{\psi=0} = \frac{3\alpha}{4\pi^2} \gamma^2 \frac{\Delta\omega}{\omega} \frac{I}{e} H_2(y), \quad (2)$$

where

$$H_2(y) = y^2 K_{2/3}^2 \left[\frac{y}{2} \right].$$

In practical units [photons / sec / mrad² / 0.1% bandwidth],

$$\left. \frac{d^2 F_{bm}(y)}{d\theta d\psi} \right|_{\psi=0} = 1.327 \times 10^{13} E^2 [\text{GeV}] I[A] H_2(y) . \quad (2a)$$

Eq. (2a), the central ($\psi = 0$) intensity of bending magnet radiation, is evaluated numerically in routine `bend_magnet.c` (see Bending Magnet in the Appendix).

The central intensity distribution integrated over the vertical angle ψ is given by

$$\frac{dF_{bm}(y)}{d\theta} = \frac{\sqrt{3}}{2\pi} \alpha \gamma \frac{\Delta\omega}{\omega} \frac{I}{e} G_1(y) , \quad (3)$$

where

$$G_1(y) = y \int_y^\infty K_{5/3}(y') dy' .$$

In practical units [photons / sec / mrad / 0.1% bandwidth],

$$\frac{dF_{bm}(y)}{d\theta} = 2.457 \times 10^{13} E[\text{GeV}] I[A] G_1(y) . \quad (3a)$$

The bending magnet flux in Eqs. (3,3a) is a constant function of the horizontal angle θ . Equation (3a) is evaluated numerically in routine `bend_magnet.c` (see Bending Magnet in the Appendix). The flux of all SR sources described in this report is plotted in Fig. 1.

The intensity of a source provides information about the angular collimation of its flux and is useful in determining the fraction of source flux which will be transmitted by an angular aperture, for example. That is, intensity is flux per unit solid angle. Central intensity is the intensity along the principal (central) ray, i.e. $\theta = \psi = 0$. The central intensity of all SR sources described in this report is plotted in Fig. 2. It is also quite useful to compute the brightness of a source, which provides information about the spatial dependence of its intensity. That is, brightness is the flux per unit solid angle per unit area. Brightness determines the fraction of source flux which can be transmitted by an optical system (slits, mirrors, grating, crystals, etc.). The brightness of all SR sources described in this report is plotted in Fig. 3.

To compute the bending magnet brightness, we must evaluate the diffraction limited bending magnet source size. In order to simplify this calculation, we neglect

the effect of electron trajectory curvature on the effective source size. The brightness expression is³

$$B_{bm} = \frac{\left. \frac{d^2 F_{bm}}{d\theta d\psi} \right|_{\psi=0}}{2\pi \Sigma_x \Sigma_y}, \quad (4)$$

where

$$\Sigma_x = \left[\epsilon_x \beta_x + \eta_x^2 \sigma_E^2 + \sigma_r^2 \right]^{1/2}, \quad \Sigma_y = \left[\epsilon_y \beta_y + \sigma_r^2 + \frac{\epsilon_y^2 + \epsilon_y \gamma_y \sigma_r^2}{\sigma_\psi^2} \right]^{1/2},$$

and

ϵ_x (ϵ_y) is the electron beam emittance in the horizontal (vertical) plane,
 β_x (β_y) is the electron beam beta function in the horizontal (vertical) plane,
 η_x is the dispersion function in the horizontal plane,
 σ_E is the rms value of the relative energy spread,
 γ_y is a Twiss parameter in the vertical plane,

$$\sigma_\psi = \frac{1}{\sqrt{2\pi}} \frac{\left. \frac{dF_{bm}}{d\theta} \right|_{\psi=0}}{\left. \frac{d^2 F_{bm}}{d\theta d\psi} \right|_{\psi=0}} \quad (5)$$

is the rms vertical opening angle of the bending magnet radiation, and
 $\sigma_r = \lambda / (4\pi\sigma_\psi)$ is the diffraction limited source size.

The effective source sizes (Σ_x and Σ_y) are photon-energy dependent via the vertical opening angle σ_ψ and the diffraction limited source size σ_r . Σ_x and Σ_y are listed in Table 1 below for $h\omega = 4h\omega_c$ (the terms involving $1/\sigma_\psi$ are most significant at the highest photon energies for each device). The central intensity is numerically divided by $2\pi\Sigma_x\Sigma_y$ in routine bend_magnet.c, yielding the bending magnet brightness (see Bending Magnet in the Appendix).

The bending magnet equations above for flux (Eqs. 2,2a), central intensity (Eqs. 3,3a), σ_ψ (Eq. 5), and brightness (Eq. 4) are all evaluated in routine bend_magnet.c (see Bending Magnet in the Appendix). This routine takes input from the terminal (or from a command file) for the values of the ring energy E , the bending radius ρ , σ_x , σ_y , η_x , σ_E , ϵ_y , and γ_y . The flux, central intensity, and brightness output is normalized to 1A beam current; the curves plotted in Figs. 1, 2, and 3 are adjusted for typical beam currents for each storage ring. The typical operating injection current values for the NSLS VUV and XRAY rings are 800 mA and 250 mA, respectively. The design injection current values for the ALS and APS rings are 400 mA and 100 mA,

respectively.

Table 1 lists the machine parameters used to evaluate the bending magnet brightness expression (Eq. (4)) for the lowest- β_y bending magnet ports on the NSLS VUV⁴, NSLS XRAY, ALS, and APS storage rings.

Table 1. Bending Magnet Machine Parameters				
	NSLS VUV	NSLS XRAY	ALS	APS
E (GeV)	0.745	2.5	1.5	7.0
ρ (m)	1.91	6.875	4.81	38.96
B (T)	1.28	1.22	1.04	0.599
ϵ_x (mrad)	1.38×10^{-7}	1.02×10^{-7}	1.0×10^{-8}	8.2×10^{-9}
β_x (m)	1.02	1.98	0.855	1.74
σ_E	8.0×10^{-4}	8.0×10^{-4}	8.0×10^{-4}	9.0×10^{-4}
η_x (m)	0.562	0.528	0.094	0.030
ϵ_y (mrad)	4.0×10^{-9}	1.02×10^{-9}	1.0×10^{-9}	8.2×10^{-10}
β_y (m)	9.70	8.18	1.46	16.73
$\alpha_y = -\beta'_y/2$	3.77	2.16	0.108	-0.834
γ_y (m ⁻¹)	0.183	0.69	0.69	0.101
σ_x (mm)	0.58	0.62	0.119	0.122
σ_y (mm)	0.197	0.091	0.0382	0.117
Σ_x (mm)	0.58	0.62	0.119	0.122
Σ_y (mm)	0.198	0.092	0.0392	0.123

In Table 1 above, $\gamma_y = \frac{1 + \alpha_y^2}{\beta_y}$, $\sigma_x = \sqrt{\epsilon_x \beta_x + \eta_x^2 \sigma_E^2}$, and $\sigma_y = \sqrt{\epsilon_y \beta_y}$; Σ_x and Σ_y are given in Eq. (4), and are listed for $h\omega = 4h\omega_c$.

The flux, central intensity, and brightness of bending magnet radiation from the NSLS VUV, NSLS XRAY, ALS, and APS storage rings are plotted in Figs. 1, 2, and 3 respectively. The bending magnet curves are labeled "VUVBM" (NSLS VUV), "XRBM" (NSLS XRAY), "ALSBM", and "APSBM".

III. Wigglers

In an ordinary planar undulator or wiggler, the stored electrons pass through a magnetic field which varies approximately sinusoidally in the vertical plane. The

resulting motion of the electrons is also approximately sinusoidal, but in the horizontal plane. Because of this periodic motion, extra factors enter into the calculations for flux and brightness. The electron motion is characterized by the deflection parameter K given by

$$K = \frac{eB_0\lambda_u}{2\pi mc} = 0.934 \lambda_u[\text{cm}] B_0[\text{T}] , \quad (6)$$

where B_0 is the amplitude and λ_u is the period of the sinusoidal magnetic field.

When K is large ($K \gg 1$), the periodic magnetic structure is called a wiggler. In this case, the flux distribution is given by $2N$ (where N is the number of magnetic periods) times the appropriate bending magnet formula [Eq. (2,2a) for central intensity, Eq. (3,3a) for flux]. However, ρ or B must be taken at the point in the path of the electron which is tangent to the direction of observation. For an horizontal angle θ ,

$$\epsilon_c(\theta) = \epsilon_{c\max} \sqrt{1 - (\theta/\delta)^2}$$

where

$$\delta = \frac{K}{\gamma} \text{ and } \epsilon_{c\max} = 0.665 E^2[\text{GeV}] B_0[\text{T}] .$$

The numerical integration of the bending magnet flux over the horizontal angular distribution is performed in routine wig_flux.c (see Wiggler Flux in the Appendix). This program expects the following input from the terminal: ring energy E , wiggler period λ_u , wiggler K , horizontal deviation half-angle, horizontal deviation angle increment for the integration, and number of magnetic poles $2N$. The resulting wiggler flux output is normalized to 1A stored beam current; the curves plotted in Figs. 1, 2, and 3 are multiplied by the typical storage ring current (in Amps). The flux plotted in Fig. 1 is integrated over the horizontal angle range $-2.5^\circ < \theta < 2.5^\circ$, which roughly matches the usable output from the wigglers described in this report.

The wiggler central intensity is simply $2N$ times the bending magnet central intensity formula (Eqs. 2,2a) with $B = B_0$, since the full magnetic field strength is felt along the central ($\theta = \psi = 0$) ray. Wiggler central intensity is calculated in routine wig_bright.c (see Wiggler Brightness in the Appendix).

The calculation of the brightness of wigglers needs to take into account the depth-of-field effects, i.e. the contribution to the apparent source size from different poles. The expression for the brightness of wigglers is³

$$B_w = \frac{d^2 F_w}{d\theta d\psi} \sum_{\pm} \sum_{n=\left\lfloor \frac{-N}{2} \right\rfloor}^{\left\lfloor \frac{N}{2} \right\rfloor} \frac{1}{2\pi} \left[\frac{\exp \left[\frac{-1}{2} \left(\frac{x_0^2}{\sigma_x^2 + z_{n\pm}^2 \sigma_x'^2} \right) \right]}{\left[\left(\sigma_x^2 + z_{n\pm}^2 \sigma_x'^2 \right) \left(\frac{\epsilon_y^2}{\sigma_\psi^2} + \sigma_y^2 + z_{n\pm}^2 \sigma_y'^2 \right) \right]^{1/2}} \right], \quad (7)$$

where

$$z_{n\pm} = \lambda_w \left[n \pm \frac{1}{4} \right], \quad \lambda_w = \text{wiggler period, and}$$

$$\sigma_\psi = \frac{1}{\sqrt{2\pi}} \frac{\left. \frac{dF_w}{d\theta} \right|_{\theta=0}}{\left. \frac{d^2 F_w}{d\theta d\psi} \right|_{\theta=\psi=0}}, \quad (8)$$

where

$$\left. \frac{dF_w}{d\theta} \right|_{\theta=0} = \text{on axis flux of the wiggler,}$$

$$\left. \frac{d^2 F_w}{d\theta d\psi} \right|_{\theta=\psi=0} = \text{on axis intensity of the wiggler, and}$$

$\sigma_x = \sqrt{\epsilon_x \beta_x}$ and $\sigma'_x = \sqrt{\epsilon_x / \beta_x}$ ($\sigma_y = \sqrt{\epsilon_y \beta_y}$ and $\sigma'_y = \sqrt{\epsilon_y / \beta_y}$) = rms transverse size and angular divergence of the electron beam in the horizontal (vertical) directions, respectively, at the center of an insertion straight section ($\eta_x = \alpha_{x,y} = 0$).

The exponential factor in Eq. (7) arises because wigglers have two source points, separated by $2x_0$, where

$$x_0 = \frac{K}{\gamma} \frac{\lambda_w}{2\pi}.$$

The summations in Eq. (7) must be performed for each photon energy because σ_ψ is photon energy dependent. The calculation of σ_ψ vs. photon energy, using Eq. (8), and wiggler brightness, using Eq. (7), are performed in routine wig_bright.c (see Wiggler Brightness in the Appendix).

The inputs to routine wig_bright.c are the ring energy E , the on-axis bending radius $\rho_{\min}(\text{m}) = 3.33 E(\text{GeV})/B_0(\text{T})$, the number of wiggler periods N , the wiggler magnetic strength K , the wiggler magnetic period λ_w , σ_x , σ_y , σ'_x , σ'_y , and the vertical beam emittance ϵ_y . The wiggler and machine parameters needed to evaluate wiggler

flux, central intensity, and brightness are listed for NSLS VUV⁴, NSLS XRAY, ALS, and APS wigglers in Table 2 below. The flux, central intensity, and brightness of wiggler radiation from NSLS U13U⁴, NSLS X17, NSLS X21 and X25, ALS W16.0, and APS wiggler A (labeled "WIGA") are plotted in Figs. 1, 2, and 3 respectively.

Table 2. Wiggler and Insertion Straight Section Parameters					
	NSLS VUV	NSLS XRAY		ALS	APS
E (GeV)	0.745	2.5		1.5	7.0
	U13U	X17	X21,X25	W16.0	Wiggler A
Wiggler parameters					
Magnet material	NdFeB	Fe	SmCo	NdFeB	NdFeB
Magnet design	Hybrid	Superconducting	Hybrid	Hybrid	Hybrid
λ_u , period length (mm)	100	174	120	160	150
2N, number of poles	45	5.5	27	33	20
Minimum gap (mm)	30.0	32.0	26.2	16.5	39.6
B _{max} (T)	0.81	5.00	1.10	2.07	1.00
K _{max}	7.56	81.26	12.33	26.3	14.01
ρ_{min} (m)	3.06	1.66	7.57	2.41	23.31
Insertion straight section parameters					
ϵ_x (mrad)	1.38×10^{-7}	1.02×10^{-7}		1.0×10^{-8}	8.2×10^{-9}
β_x (m)	11.1	1.685		11.2	13.377
ϵ_y (mrad)	4.0×10^{-9}	1.02×10^{-9}		1.0×10^{-9}	8.2×10^{-10}
β_y (m)	5.84	0.354		4.10	10.00
σ_x (mm)	1.24	0.410		0.330	0.331
σ_y (mm)	0.153	0.019		0.063	0.091
σ_x (mrad)	0.112	0.246		0.030	0.025
σ_y (mrad)	0.026	0.054		0.016	0.009

IV. Undulators

In an undulator, the deflection parameter K is moderate ($K \leq 1$) and interference of the radiation from different periods produces sharp peaks at a fundamental ($n=1$) and its harmonics ($n=2,3,\dots$). The peak wavelength of the fundamental, λ_1 , is given by

$$\lambda_1 = \frac{\lambda_u}{2\gamma^2} \left(1 + \frac{K^2}{2}\right), \quad (9)$$

where λ_u is the undulator period length.

Our main interest is the intense central cone of radiation. An approximate formula for flux integrated over the central cone is (for the odd harmonics)³

$$F_u(K, \omega) = \pi \alpha N \frac{\Delta \omega}{\omega} \frac{I}{e} Q_n(K), \quad n=1,3,5... \quad (10)$$

where

$$Q_n(K) = \left(1 + \frac{K^2}{2}\right) \frac{F_n(K)}{n}, \quad n=1,3,5... \text{ and}$$

$$F_n(K) = \frac{K^2 n^2}{(1 + K^2/2)^2} \left\{ J_{\frac{n-1}{2}} \left[\frac{nK^2}{4(1 + K^2/2)} \right] - J_{\frac{n+1}{2}} \left[\frac{nK^2}{4(1 + K^2/2)} \right] \right\}^2.$$

In practical units [photons / sec / 0.1% bw],

$$F_u(K) = 1.431 \times 10^{14} N Q_n(K) I[A]. \quad (10a)$$

Eq. (10a) is evaluated numerically in routine undulator.c (see Undulator in the Appendix), and then multiplied by the typical storage ring current (in Amps) to produce undulator central cone flux. The inputs to routine undulator.c are the ring energy E , the undulator magnetic period length λ_u , the number of undulator periods N , the minimum and maximum of the usable range of K , σ_x , σ_y , σ'_x , and σ'_y .

The central intensity of an undulator is approximated by dividing the central cone flux by the rms effective angular divergence Σ'_x (Σ'_y) in the horizontal (vertical) directions. Thus, the central intensity is

$$\left. \frac{d^2 F_u}{d\theta d\psi} \right|_{\psi=\theta=0} = \frac{F_u}{2\pi \Sigma'_x \Sigma'_y}, \quad (11)$$

where

$$\Sigma'_x = \sqrt{\sigma'^2_x + \sigma'^2_r}, \quad \Sigma'_y = \sqrt{\sigma'^2_y + \sigma'^2_r}, \quad \text{and } \sigma'_r = \sqrt{\frac{\lambda}{L}}.$$

Eq. (11) is evaluated numerically in routine undulator.c to produce undulator central intensity (see Undulator in the Appendix).

To calculate the brightness of an undulator, the central intensity must be divided

by the rms effective source size, Σ_x (Σ_y), in the horizontal (vertical) directions. Thus, the undulator brightness is given by

$$B_u \equiv \frac{F_u}{(2\pi)^2 \Sigma_x \Sigma_y \Sigma'_x \Sigma'_y}, \quad (12)$$

where

$$\Sigma_x = \sqrt{\sigma_x^2 + \sigma_r^2}, \quad \Sigma_y = \sqrt{\sigma_y^2 + \sigma_r^2}, \quad \text{and} \quad \sigma_r = \frac{1}{4\pi} \sqrt{\lambda L}.$$

Eq. (12) is evaluated numerically in routine undulator.c to produce undulator brightness (see Undulator in the Appendix).

The undulator and machine parameters needed to evaluate Eqs. (10a), (11), and (12) are listed for NSLS VUV, NSLS XRAY, ALS, and APS undulators in Table 3 below.

Table 3. Undulator and Insertion Straight Section Parameters							
	NSLS VUV		NSLS XRAY	ALS			APS
E (GeV)	0.745		2.5	1.5			7.0
	U5U	U13U	X1	U3.9	U5.0	U8.0	UA
Undulator parameters							
Magnet material	NdFeB	NdFeB	SmCo	NdFeB	NdFeB	NdFeB	NdFeB
Magnet design	Hybrid	Hybrid	Hybrid	Hybrid	Hybrid	Hybrid	Hybrid
λ_u , period length (mm)	75	100	80	39	50	80	33
2N, number of poles	55	45	70	228	178	110	300
Max, min gap (mm)	120,33.5	120,31.4	98,33	33.1,15.5	47.6,15.6	97.0,20.4	29.0,10.4
Min, max B (T)	0.05,0.48	0.07,0.81	0.03,0.31	0.14,0.58	0.11,0.82	0.07,1.04	0.13,0.81
Min, max K	0.35,3.4	0.65,7.6	0.2,2.3	0.5,2.1	0.5,3.8	0.5,7.8	0.4,2.5
Insertion straight section parameters							
ϵ_x (mrad)	1.38×10^{-7}		1.02×10^{-7}	1.0×10^{-8}			8.2×10^{-9}
β_x (m)	11.1		1.685	11.2			13.377
ϵ_y (mrad)	4.0×10^{-9}		1.02×10^{-9}	1.0×10^{-9}			8.2×10^{-10}
β_y (m)	5.84		0.354	4.10			10.00
σ_y (mm)	1.240		0.410	0.330			0.331
σ_x (mm)	0.153		0.019	0.063			0.091
σ_y (mrad)	0.112		0.246	0.030			0.025
σ_x (mrad)	0.026		0.054	0.016			0.009
Min, max σ_r (μ)	8.0,38.4	9.4,44.3	3.4,10.6	4.3,12.8	5.1,27.2	6.2,39.1	0.9,3.3
Min, max σ_r (μ rad)	48.8,223.8	52.5,247.4	15.0,47.5	11.4,33.5	13.0,69.8	15.9,100.8	2.2,8.5

The flux, central intensity, and brightness of undulator radiation from NSLS U5U⁴, NSLS U13U⁴, NSLS X1, ALS U3.9, ALS U5.0, ALS U8.0, and APS undulator A (labeled "UA") are plotted in Figs. 1, 2, and 3 respectively.

Conclusions

The programs and calculations presented here can be used to calculate the flux, central intensity, vertical opening angle, effective source sizes (horizontal and vertical), and brightness of bending magnets and ordinary planar undulators and wigglers on any synchrotron storage ring. Listings of the routines used can be found in the Appendix.

The flux, central intensity, and brightness of bending magnets, wigglers, and undulators at the NSLS VUV, NSLS XRAY, ALS, and APS storage rings are plotted in Figs. 1, 2, and 3 respectively.

Figure Captions

1. Spectral flux as a function of photon energy for selected undulators, wigglers, and bending magnets at the NSLS, ALS, and APS. The NSLS devices are all in operation; the ALS and APS devices are planned. The wiggler and bending magnet flux curves represent the vertically-integrated flux within a 5 mrad horizontal collection angle. Each undulator curve is the locus of narrow peaks of radiation, tuned by altering the undulator magnetic strength parameter K over the range of values listed in Table 3, and represents the envelope of the first, third, and fifth harmonics. All of the curves correspond to standard (or planned) injection current values for each storage ring: NSLS VUV: 0.8A, NSLS XRAY: 0.25A, ALS: 0.4A, and APS: 0.1A.
2. Spectral central intensity as a function of photon energy for selected undulators, wigglers, and bending magnets at the NSLS, ALS, and APS. The NSLS devices are all in operation; the ALS and APS devices are planned. Each undulator curve is the locus of narrow peaks of radiation, tuned by altering the undulator magnetic strength parameter K over the range of values listed in Table 3, and represents the envelope of the first, third, and fifth harmonics. All of the curves correspond to standard (or planned) injection current values for each storage ring: NSLS VUV: 0.8A, NSLS XRAY: 0.25A, ALS: 0.4A, and APS: 0.1A.
3. Spectral brightness as a function of photon energy for selected undulators, wigglers, and bending magnets at the NSLS, ALS, and APS. The NSLS devices are all in operation; the ALS and APS devices are planned. Each undulator curve is the locus of narrow peaks of radiation, tuned by altering the undulator magnetic strength parameter K over the range of values listed in Table 3, and represents the envelope of the first, third, and fifth harmonics. All of the curves correspond to standard (or planned) injection current values for each storage ring: NSLS VUV: 0.8A, NSLS XRAY: 0.25A, ALS: 0.4A, and APS: 0.1A.

References

1. J. Schwinger, Phys. Rev. **75**, 1912 (1949).
2. Vaclav O. Kostroun, Nucl. Instrum. Meth. **172**, 371 (1980).
3. Lawrence Berkeley Laboratory, A Summary of Capabilities and Characteristics of the Advanced Light Source (1989).
4. $\epsilon_y = 4.0 \times 10^{-9}$ mrad (approximately 3 percent coupling) is a typical operating value for the NSLS VUV ring. This value of the vertical emittance provides a useful compromise between small vertical beam size and long lifetime. In high brightness mode, ϵ_y can be made as small as 1.4×10^{-9} mrad.

APPENDIX

Bending Magnet

```

/* Bending Magnet Flux, Central Intensity, Vertical Source Size,    */
/*      Brightness: bend_magnet.c                                   */

#include <stdio.h>
#include <math.h>
#include "io.c"

#define HC 1.239854e-3 /* hc [eV*mm] */

main(argc,argv)
int argc;
char *argv[];
{
    double ering = 0.745; /* storage ring energy in GeV */
    double rad = 1.91; /* bending magnet radius in m */
    double gam;
    double sigx = 0.375; /* sigma x (mm) */
    double sigy = 0.197; /* sigma y (mm) */
    double emitty = 0.005; /* y emittance (mm*mrad) */
    double DispX = 562.0; /* dispersion fcn in x direction (mm) */
    double sigmae = 8.e-4; /* rms relative electron energy spread */
    double gammay = 0.183e-3; /* third betatron fcn in y direction (mm-1) */
    double pi;
    double ecrit; /* critical energy in eV */
    double ah=0.5, prec1=0.00001, nu;
    double xi, p, term, rh;
    double x, emin, emax, deltae, fl, ci, vsig, bri;
    double sigmar, sigmax, sigmay;
    int nints, islog, k;
    int argptr=1;

    if( argc == 1 ) {
        printf("Ring    energy (GeV)    radius(m)    B(T)    Ecrit(eV)\n");
        printf("-----\n");
        printf("NSLS VUV    0.745        1.91        1.28        472.4\n");
        printf("Aladdin    0.8          2.08        1.28        544.8\n");
        printf("LBL-ALS    1.5          4.81        1.04        1556.1\n");
        printf("NSLS XRAY   2.5          6.875       1.22        5070.6\n");
        printf("SSRL       3.0          12.61       0.79        4728.2\n");
        printf("CHESS      5.5          31.75       0.58        11667.\n");
        printf("ANL - APS   7.0          38.96       0.60        19551.\n");
    }

    if( argptr < argc ) sscanf( argv[argptr++], "%lf", &ering );
    else get_dnum("Enter ring energy (GeV)", &ering);
    if( argptr < argc ) sscanf( argv[argptr++], "%lf", &rad );
    else get_dnum("Enter bending magnet radius (m)", &rad);
    if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigx );
    else get_dnum("sigma x (=sqrt(epsx*betax)) (mm)", &sigx);
    if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigy );
    else get_dnum("sigma y (=sqrt(epsy*betay)) (mm)", &sigy);

```

```
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &Dispx );
else get_dnum("x dispersion (mm)", &Dispx );
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigmae );
else get_dnum("sigmaE / E", &sigmae);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emitty );
else get_dnum("y emittance (mm*mmrad)", &emitty);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &gammay );
else get_dnum("Twiss parameter gamma y (mm^-1)", &gammay);

printf("Ering = %f GeV, radius= %f m\n", ering, rad);
printf("sigma x = %f mm, sigma y = %f mm\n", sigx, sigy);
printf("x-dispersion = %f mm, (sigma E)/E = %f\n", Dispx, sigmae);
printf("y emittance = %f mm*mmrad, gamma y = %f (mm^-1)\n", emitty, gammay);

ecrit = 665.0*ering*ering*3.33*ering/rad;
printf("Ecrit = %f eV\n", ecrit);

gam = ering * 1000.0 / 0.511;      /* convert to gamma = Ek/m0c2 */
emitty /= 1e3;                    /* convert to mm*rad */

pi = 4. * atan(1.0);

if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emin );
else get_dnum( "Emin (eV)?", &emin );
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emax );
else get_dnum( "Emax (eV)?", &emax );
if( argptr < argc ) sscanf( argv[argptr++], "%d", &nints );
else get_inum( "# of intervals?", &nints );
if( argptr < argc ) sscanf( argv[argptr++], "%d", &islog );
else get_inum( "logarithmic intervals (1/0)?", &islog);
printf("emin= %g eV, emax= %g eV, %d intervals, %s\n", emin, emax, nints,
       islog ? "log" : "linear");

if( islog ) { emax = log(emax); emin = log(emin); }
deltae = (emax - emin) / (double)nints;

printf("\nBending magnet central intensity:\n");
printf("y-units will be: photons/sec/0.1%bw/A/mrad**2\n");

printf("\nBending magnet flux:\n");
printf("y-units will be: photons/sec/mrad/0.1%bw/A\n");

printf("\nBending magnet vertical opening angle (mrad)\n");

printf("\nBending magnet diffraction-limited source size (mm)\n");
printf("\nBending magnet effective vertical source size (mm)\n");
printf("\nBending magnet effective horizontal source size (mm)\n");

printf("\nBending magnet brightness.\n");
printf("Brightness = central intensity / (2*pi*(effsigh)*(effsigv)),\n");
printf("where effsig[h,v] are effective [h,v] source size sigma's.\n");
printf("y-units will be: photons/sec/0.1%bw/A/mrad**2/mm**2\n");
printf("nhv(eV), central intensity, flux, sigmapsi, sigmar, sigmax, sigmay, brightness.\n");
/*-----*/

for( x=emin; x<=emax; x+=deltae ) {      /* x loop */
```

```
/* "bending magnet" central intensity calculation */
/*
CALCULATE BESSEL FUNCTION K(2/3) FROM SERIES BY KOSTROUN
*/
nu = 2.0/3.0;
xi = (islog ? exp(x):x)/ecrit;

p = 1.0;
ci = 0.0;
do {
    term = exp(-xi/2.0*cosh(p*ah)) * cosh(nu*p*ah);
    ci += term;
    p += 1.0;
} while( term > prec1 * ci );
ci += 0.5 * exp(-xi/2.0);
ci *= ah;

/* photons/sec/0.1%bw/Amp/mrad**2 */
ci *= xi*xi*ci;
ci *= 1.327e13*ering*ering;

printf("%g %g ", islog ? exp(x):x, ci);

/*-----*/
/* "bending magnet" flux calculation */
/*
CALCULATE integral of BESSEL FUNCTION K(nu) FROM SERIES
BY KOSTROUN
*/
nu = 5.0/3.0;
fl = ah * exp(-xi) / 2.0;
for( k=1; k<=500; k++ ) {
    rh = (double)k*ah;
    term = ah * exp(-xi*cosh(rh)) * cosh(nu*rh) / cosh(rh);
    fl += term;
    if(term < prec1) { break; }
} /* end for k */

/* photons/sec/mrad/0.1%bw/A */
fl *= 2.457e13*ering*xi;

printf("%g ", fl);

/*-----*/
/* vertical opening angle : mrad */
vsig = fl / ci / sqrt(2.0*pi);

printf("%g ", vsig);

/*-----*/
/* Bending magnet source size calculation */

vsig /= 1e3; /* convert from mrad to rad */
sigmar = HC / (islog ? exp(x):x) / 4.0 / pi / vsig;
printf("%g ", sigmar); /* diffraction-limited source size : mm */

sigmax = sigx * sigx;
```

```

sigmax += Dispx * Dispx * sigmae * sigmae;
sigmay += sigmar * sigmar;
sigmax = sqrt(sigmax); /* effective sigma x (mm) */
printf("%g ", sigmax); /* effective horizontal source size: mm */

sigmay = sigy * sigy;
sigmay += sigmar * sigmar;
sigmay += (emitty*(emitty + gammay*sigmar*sigmar))/vsig/vsig;
sigmay = sqrt(sigmay); /* effective sigma y (mm) */
printf("%g ", sigmay); /* effective vertical source size: mm */

bri = ci / 2.0 / pi / sigmax / sigmay;
/* photons/sec/0.1%bw/Amp/mm**2/mrad**2 */

printf("%g\n", bri);

} /* end x for loop */

}

```

Wiggler Flux

```

/* Wiggler Flux : wig_flux.c */

/*
  CALCULATE integral of BESSEL FUNCTION K(nu) FROM SERIES
  BY KOSTROUN
*/

#include <stdio.h>
#include <math.h>
#include "io.c"

double ering = 0.745; /* storage ring energy in GeV */
double gam; /* ring energy in electron rest mass units */
double period = 10.0; /* wiggler period in cm */
double kwig = 8.0; /* wiggler magnetic strength parameter K */
double ecmax; /* maximum critical energy in eV */
double ah=0.5, prec1=0.00001, nu;
double delta; /* 1.2 K / gamma */
double horzang = 2.5; /* horiz. deviation half-angle to collect (mrad) */
double deltang = 0.1; /* horiz. deviation angle increment (mrad) */
double npoles = 45.0; /* # of poles */

main(argc,argv)
int argc;
char *argv[];
{

double x, emin, emax, deltae, y;
double angle;
double xi, term, rh;
double ecrit, besint;
int nints, islog, k;
int argptr=1;

```

nu = 5.0/3.0;

```
if( argc == 1 ) {
    printf("\nSynchrotron radiation flux \n");
    printf("Ring      energy (GeV)  period(cm)  B(T)    Ecrit(eV)\n");
    printf("-----\n");
    printf("NSLS-U13U    0.745      10.0      0.86      317.4\n");
    printf("ALS-W16.0    1.5        16.0      2.07      3097.2\n");
    printf("NSLS-X21,X25  2.5        12.0      1.10      4571.9\n");
    printf("NSLS-X17     2.5        17.4      5.00      20781.\n");
    printf("APS-WIGA     7.0        15.0      1.00      32585.\n");
}
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &ering );
else get_dnum("Enter ring energy (GeV)", &ering);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &period );
else get_dnum("Enter wiggler period (cm)", &period);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &kwig);
else get_dnum("Enter wiggler K", &kwig);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &horzang);
else get_dnum("Enter horiz. deviation half-angle to collect (mrad)", &horzang);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &deltang);
else get_dnum("Enter horiz. angle increment for calc (mrad)", &deltang);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &npoles);
else get_dnum("Enter # of poles (2N)", &npoles);

printf("Ering= %f GeV, wiggler period= %f cm, K= %f\n",ering,period,kwig);
printf("horiz. half-angle to collect= %f mrad\n", horzang);
printf("horiz. angle increment for calc= %f mrad\n", deltang);
printf("# of poles = %f\n", npoles);

ecmax = 665.0 * ering * ering * kwig / 0.934 / period;
printf("Ecrit (max)= %f eV\n", ecmax);

gam = 1000.0 * ering / 0.511;
delta = 1.e3*kwig/gam;
printf("K= %f, gamma= %f, K/gamma= %f mrad\n", kwig, gam, delta);

if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emin);
else get_dnum( "Emin (eV)?", &emin );
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emax);
else get_dnum( "Emax (eV)?", &emax );
if( argptr < argc ) sscanf( argv[argptr++], "%d", &nints);
else get_inum( "# of intervals?", &nints );
if( argptr < argc ) sscanf( argv[argptr++], "%d", &islog);
else get_inum( "logarithmic intervals (1/0)?", &islog);
printf("emin=%f, emax=%f, %d intervals, %s\n", emin, emax, nints,
        islog ? "logarithmic" : "linear");

if( islog ) { emax = log(emax); emin = log(emin); }
deltae = (emax - emin) / (double)nints;

printf("\nWiggler flux:\n");
printf("y-units will be: photons/sec/%f mrad/0.1%bw/A\n", 2.*horzang);
printf("nhv(eV), flux\n");

for( x=emin; x<=emax; x+=deltae ) { /* x loop */
```

```

y = 0.0;
for(angle=0.0; angle <= horzang; angle+=deltang) {
    if( angle >= delta ) continue;
    ecrit = angle / delta;
    ecrit = eamax * sqrt(1.0 - ecrit * ecrit);
/*    printf("E crit = %f eV\n", ecrit);*/
    if(!islog) xi = exp(x)/ecrit; else xi = x/ecrit;
    besint = ah * exp(-xi) / 2.0;
    for(k=1; k<=500; k++) {
        rh = (double)k*ah;
        term = ah * exp(-xi*cosh(rh)) * cosh(nu*rh) / cosh(rh);
        besint += term;
        if( term < prec1 ) break;
    }    /* end for k */

    if(angle != 0.0) y += 2.*deltang*xi*besint; /* *deltang cuz' formula */
                                   /* is per mrad */
                                   /* *2 to count both + and - horiz. */
                                   /* deviation angles */
    else y += deltang*xi*besint;
}    /* end for angle */

y *= 2.457e13*ering; /* photons/sec/0.1%bw/horzang/A */;
y *= npoles;

printf("%g %g\n", islog ? exp(x):x, y);
}

}

```

Wiggler Brightness

```

/* Wiggler Central Intensity, Central Flux, Vertical Source Size, */
/*    Brightness : wig_bright.c */

#include <stdio.h>
#include <math.h>
#include "io.c"

main(argc,argv)
int argc;
char *argv[];
{
    double N = 22.5; /* # of periods */
    double K = 8; /* wiggler K */
    double ering = 0.745; /* storage ring energy in GeV */
    double gam;
    double wiglam = 100.0; /* wiggler period (mm) */
    double sigx = 1.24; /* sigma x (mm) */
    double sigy = 0.17; /* sigma y (mm) */
    double sigxp = 0.112; /* sigma x prime (mrad) */
    double sigyp = 0.029; /* sigma y prime (mrad) */
    double emitty = 0.005; /* y emittance (mm*mrad) */
    double pi;
    double x0; /* dist (mm) betw two lobes of wiggler source */
}

```



```

double rad = 2.90;      /* bending magnet radius in m */
double ecrit;          /* critical energy in eV */
double ah=0.5, prec1=0.00001, nu;
double xl, p, term, rh;
double x, emin, emax, deltae, fl, ci, vsig, bri;
double dn, zn, sigma_x, y1;
int nints, islog, k;
int i, n, nmin, nmax;
int argptr=1;

if( argc == 1 ) {
    printf("\nSynchrotron radiation flux\n");
    printf("Ring      energy (GeV)  period(cm)  B(T)    Ecrit(eV)\n");
    printf("-----\n");
    printf("NSLS-U13U    0.745      10.0      0.86     317.4\n");
    printf("ALS-W16.0    1.5        16.0      2.07     3097.2\n");
    printf("NSLS-X21,X25  2.5        12.0      1.10     4571.9\n");
    printf("NSLS-X17     2.5        17.4      5.00     20781.\n");
    printf("APS-WIGA     7.0        15.0      1.00     32585.\n");
}

if( argptr < argc ) sscanf( argv[argptr++], "%lf", &ering );
else get_dnum("Enter ring energy (GeV)", &ering);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &rad );
else get_dnum("Enter bending magnet radius (m)", &rad);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &N );
else get_dnum("# periods", &N);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &K );
else get_dnum("Wiggler K", &K);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &wiglam );
else get_dnum("Wiggler period (mm)", &wiglam);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigx );
else get_dnum("sigma x (mm)", &sigx);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigy );
else get_dnum("sigma y (mm)", &sigy);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigxp );
else get_dnum("sigma x prime (mrad)", &sigxp);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigyp );
else get_dnum("sigma y prime (mrad)", &sigyp);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emitty );
else get_dnum("y emittance (mm*mrad)", &emitty);

printf("Ering = %f GeV, radius= %f m\n", ering, rad);
printf("# periods = %f\n", N);
printf("K = %f, wiggler period = %f mm\n", K, wiglam);
printf("sigma x = %f mm, sigma y = %f mm\n", sigx, sigy);
printf("sigma x prime = %f mrad, sigma y prime = %f mrad\n", sigxp, sigyp);
printf("y emittance = %f mm*mrad\n", emitty);

ecrit = 665.0*ering*ering*3.33*ering/rad;
printf("Ecrit = %f eV\n", ecrit);

gam = ering * 1000.0 / 0.511;      /* convert to gamma = Ek/m0c2 */
sigxp /= 1e3;                      /* convert to rad */
sigyp /= 1e3;                      /* convert to rad */
emitty /= 1e3;                     /* convert to mm*mrad */

pi = 4. * atan(1.0);

```

```
x0 = K * wlgam / gam / 2.0 / pi;
printf("x0 = K * wlgam / gam / 2 / pi = %g\n", x0);

nmax = (int)N;
nmin = -1 * nmax;
printf("nmin= %d, nmax = %d\n", nmin, nmax);

if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emin );
else get_dnum( "Emin (eV)?", &emin );
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emax );
else get_dnum( "Emax (eV)?", &emax );
if( argptr < argc ) sscanf( argv[argptr++], "%d", &nints );
else get_inum( "# of intervals?", &nints );
if( argptr < argc ) sscanf( argv[argptr++], "%d", &islog );
else get_inum( "logarithmic intervals (1/0)?", &islog );
printf("emin= %g eV, emax= %g eV, %d intervals, %s\n", emin, emax, nints,
       islog ? "log" : "linear");

if( islog ) { emax = log(emax); emin = log(emin); }
deltae = (emax - emin) / (double)nints;

printf("\nWiggler central intensity:\n");
printf("y-units will be: photons/sec/0.1%bw/A/mrad**2\n");

printf("\nWiggler central flux:\n");
printf("y-units will be: photons/sec/mrad/0.1%bw/A\n");

printf("\nWiggler vertical opening angle (mrad)\n");

printf("\nWiggler effective source area = 2*pi*effsigh*effsigv (mm**2)\n");
printf("where effsig[h,v] are effective [h,v] source size sigma's.\n");

printf("\nWiggler brightness.\n");
printf("Brightness = central intensity / (2*pi*(effsigh)*(effsigv)),\n");
printf("y-units will be: photons/sec/0.1%bw/A/mrad**2/mm**2\n");
printf("\nhv(eV), central intensity, central flux, vert. opening angle, eff. source area, brightness.\n");
/*-----*/

for( x=emin; x<=emax; x+=deltae ) { /* x loop */

/*-----*/
/* wiggler central intensity calculation */
/*
CALCULATE BESSEL FUNCTION K(2/3) FROM SERIES BY KOSTROUN
*/
nu = 2.0/3.0;
if(islog) xi = exp(x)/ecrit;
else xi = x/ecrit;

p = 1.0;
ci = 0.0;
do {
term = exp(-xi/2.0*cosh(p*ah)) * cosh(nu*p*ah);
ci += term;
p += 1.0;
} while( term > prec1 * ci );
ci += 0.5 * exp(-xi/2.0);
```

```

ci *= ah;

/* photons/sec/0.1%bw/Amp/mrad**2 */
ci *= xi*xi*ci;
ci *= 1.327e13*ering*ering;

printf("%g %g ", islog ? exp(x):x, ci);

/*-----*/
/* wiggler central flux calculation */
/*
   CALCULATE integral of BESSEL FUNCTION K(nu) FROM SERIES
   BY KOSTROUN
*/
nu = 5.0/3.0;
fl = ah * exp(-xi) / 2.0;
for( k=1; k<=500; k++ ) {
    rh = (double)k*ah;
    term = ah * exp(-xi*cosh(rh)) * cosh(nu*rh) / cosh(rh);
    fl += term;
    if(term < prec1) { break; }
} /* end for k */

/* photons/sec/mrad/0.1%bw/A */
fl *= 2.457e13*ering*xi;

printf("%g ", fl);

/*-----*/
/* vertical opening angle : mrad */
vsig := fl / ci / sqrt(2.0*pi);

printf("%g ", vsig);

/*-----*/
/* Wiggler brightness calculation */

sigmaxy = 0;
for( i = -1; i <= 1; i += 2 ) {
    for( n = nmin; n <= nmax; n += 2 ) {
        dn = (double)n; dn /= 2.0;
        /* printf("i= %d, n= %f\n", i, dn); */
        y1 = emitty * emitty / vsig / vsig;
        y1 += sigy * sigy;
        zn = wiglam * (dn + (double)i*0.25);
        y1 += zn * zn * sigyp * sigyp;
        y1 *= sigx * sigx + zn * zn * sigxp * sigxp;
        y1 = 0.5 / pi / sqrt(y1);
        y1 *= exp( -0.5 * x0 * x0 / (sigx*sigx + zn*zn*sigxp*sigxp) );
        sigmaxy += y1;
    } /* end for n */
} /* end for i */

printf("%g ", 1./sigmaxy); /* effective source area : mm**2 */

bri = ci * sigmaxy; /* brightness: photons/sec/0.1%bw/Amp/mm**2/mrad**2 */
printf("%g\n", bri);

```

```
    } /* end x for loop */  
}
```

Undulator

```
/* Undulator Central Cone Flux, Central Intensity, Vertical Source Size, */  
/*      Brightness: undulator.c */
```

```
/* The formula for central cone flux from undulator */  
/* is  $F = 1.431e14 * N * Q_n * I[A]$ , */  
/* where N is # of periods, I is ring current in Amps, */  
/*  $Q_n(K) = (1 + K^{**2}/2)F_n/n$  */  
/* where n is harmonic number and */  
/*  $F_n$  is  $(nK/(1+K^{**2}/2))^{**2} * \text{sum of square of } J((n-1)/2) - J((n+1)/2)$ , */  
/* where J's are Bessel fcns, argument is  $n/4 * K^{**2}/(1+K^{**2}/2)$  */
```

```
#define HC 12398.54
```

```
#include <stdio.h>  
#include <math.h>  
#include "io.c"
```

```
double period = 10.0; /* wiggler period in cm */  
double gam; /* ring energy in electron rest mass units */  
double N = 22.5; /* # of undulator periods */  
double Kmin = 0.5, Kmax = 3.0; /* min, max values of K for calculation */
```

```
main(argc,argv)  
{  
    int argc;  
    char *argv[];  
  
    double ering = 0.745; /* storage ring energy in GeV */  
    double sigx = 1.24; /* sigma x (mm) */  
    double sigy = 0.170; /* sigma y (mm) */  
    double sigpx = 0.112; /* sigma x prime (mrad) */  
    double sigpy = 0.029; /* sigma y prime (mrad) */  
    double pi;  
    double xi;  
    double x, emin, emax, deltae, fl, vopen, ci, vsig, bri;  
    double sigmarp, sigmar;  
    double Sigmaxp, Sigmax, Sigmayp, Sigmay;  
    double pfl[3], flux();  
    int nints, islog, k;  
    int argptr=1;  
  
    if( argptr < argc ) sscanf( argv[argptr++], "%lf", &ering );  
    else get_dnum("Enter ring energy (GeV)", &ering);  
    if( argptr < argc ) sscanf( argv[argptr++], "%lf", &period );  
    else get_dnum("Enter undulator period (cm)", &period);  
    if( argptr < argc ) sscanf( argv[argptr++], "%lf", &N );  
    else get_dnum("Enter # periods", &N);  
    if( argptr < argc ) sscanf( argv[argptr++], "%lf", &Kmin );  
    else get_dnum("Enter minimum K for calc", &Kmin);  
    if( argptr < argc ) sscanf( argv[argptr++], "%lf", &Kmax );
```

```

else get_dnum("Enter maximum K for calc", &Kmax);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigx );
else get_dnum("sigma x (mm)", &sigx);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigy );
else get_dnum("sigma y (mm)", &sigy);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigpx );
else get_dnum("sigma x prime (mrad)", &sigpx);
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &sigpy );
else get_dnum("sigma y prime (mrad)", &sigpy);

printf("Ering= %f GeV, undulator period= %f cm\n",ering,period);
printf("# periods = %f, ", N);
printf("%f < K < %f\n", Kmin, Kmax);
printf("sigma x = %f mm, sigma y = %f mm\n", sigx, sigy);
printf("sigma x prime = %f mrad, sigma y prime = %f mrad\n", sigpx, sigpy);

gam = ering * 1000.0 / 0.511;      /* convert to gamma = Ek/m0c2 */
printf("gamma= %f\n", gam);

pi = 4. * atan(1.0);

if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emin );
else get_dnum( "Emin (eV)?", &emin );
if( argptr < argc ) sscanf( argv[argptr++], "%lf", &emax );
else get_dnum( "Emax (eV)?", &emax );
if( argptr < argc ) sscanf( argv[argptr++], "%d", &nints );
else get_inum( "# of intervals?", &nints );
if( argptr < argc ) sscanf( argv[argptr++], "%d", &islog );
else get_inum( "logarithmic intervals (1/0)?", &islog);
printf("emin= %g eV, emax= %g eV, %d intervals, %s\n", emin, emax, nints,
       islog ? "log" : "linear");

if( islog ) { emax = log(emax); emin = log(emin); }
deltae = (emax - emin) / (double)nints;

printf("\nUndulator radiation central cone flux.\n");
printf("y-units will be: photons/sec/central cone/0.1%bw/A\n");

printf("\nUndulator diffraction limited opening angle (microrad)\n");
printf("\nUndulator horizontal opening angle (microrad)\n");
printf("\nUndulator vertical opening angle (microrad)\n");

printf("\nUndulator radiation central intensity:\n");
printf("Central intensity / (2*pi*(effsighp)*(effsigvp)),\n");
printf("where effsig[h,v]p are effective [h,v] source opening angle sigma's.\n");
printf("y-units will be: photons/sec/0.1%bw/A/mrad**2\n");

printf("\nUndulator diffraction limited source size (micron)\n");
printf("\nUndulator horizontal source size (micron)\n");
printf("\nUndulator vertical source size (micron)\n");

printf("\nUndulator brightness.\n");
printf("Brightness = central intensity / (2*pi*(effsigh)*(effsigv)),\n");
printf("where effsig[h,v] are effective [h,v] source size sigma's.\n");
printf("ny-units will be: photons/sec/0.1%bw/A/mrad**2/mm**2\n");

printf("\nhv(eV), flux, sigmar', sigmax', sigmay', central intensity, sigmar, sigmax, sigmay, brightness.\n");

```

```

/*-----*/

for( x=emin; x<=emax; x+=deltae ) { /* x loop */

/* Undulator radiation central cone flux calculation */

    xi = (islog ? exp(x):x);

    pfl[0] = flux(1,xi);
    pfl[1] = flux(3,xi);
    pfl[2] = flux(5,xi);
    if( pfl[0] > pfl[1] ) fl = pfl[0]; else fl = pfl[1];
    if( pfl[2] > fl ) fl = pfl[2];

    printf("%g %g ", xi, fl);

/*-----*/
/* opening angles (mrad) */
    /* diffraction limited opening angle */
    sigmarp = sqrt( HC/xi/N/period/100.0 );
    printf("%g ", 1.e3*sigmarp); /* microrad */
    /* eff sigma x prime */
    Sigmaxp = sqrt( sigpx*sigpx + sigmarp*sigmarp );
    printf("%g ", 1.e3*Sigmaxp); /* microrad */
    /* eff sigma y prime */
    Sigmayp = sqrt( sigpy*sigpy + sigmarp*sigmarp );
    printf("%g ", 1.e3*Sigmayp); /* microrad */
/*-----*/
/* central intensity */
    ci = fl / 2.0 / pi / Sigmaxp / Sigmayp;
    printf("%g ", ci);

/*-----*/
/* source sizes (mm) */
    /* diffraction limited source size */
    sigmar = sqrt( 1.e-7*HC/xi*N*period*10.0/4.0/pi/4.0/pi );
    printf("%g ", 1.e3*sigmar); /* microns */
    /* eff sigma x */
    Sigmax = sqrt( sigx*sigx + sigmar*sigmar );
    printf("%g ", 1.e3*Sigmax); /* microns */
    /* eff sigma y */
    Sigmay = sqrt( sigy*sigy + sigmar*sigmar );
    printf("%g ", 1.e3*Sigmay); /* microns */

/*-----*/
/* brightness */
    bri = ci / 2.0 / pi / Sigmax / Sigmay;

    printf("%g\n", bri);

} /* end x for loop */

}

double flux(nharmon,phenergy)
int nharmon;
double phenergy;

```

```
{
double K, fact, arg, y;

fact = (HC / phenergy) * 2.0e-8 / period * (double)nharmon * gam * gam;
K = sqrt( 2.0 * (fact - 1.0));
/* printf("E= %f, lambda= %f, K= %fn", x, HC/x, K);*/
if( K < Kmin | K > Kmax ) { return( 0 ); }

else {
    arg = (double)nharmon * K * K / 4.0 / fact;
    y = K * (double)nharmon / fact;
    y *= jn( (nharmon-1)/2, arg ) - jn( (nharmon+1)/2, arg );
    y *= y;
    y *= 1.431e14 * (double)N * fact / (double)nharmon;
    /* photons/sec/0.1%bw/A in central cone */
    return( y );
}
}
```

END

**DATE
FILMED**

3 / 17 / 92